

日 本 国 特 許 庁  
JAPAN PATENT OFFICE

31. 3. 2004

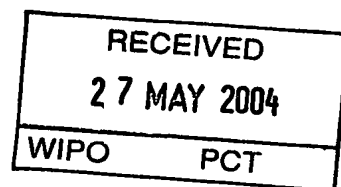
別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日  
Date of Application: 2 0 0 3 年 4 月 9 日

出 願 番 号  
Application Number: 特 願 2 0 0 3 - 1 0 5 7 6 2  
[ST. 10/C]: [ J P 2 0 0 3 - 1 0 5 7 6 2 ]

出 願 人  
Applicant(s): 財団法人北九州産業学術推進機構

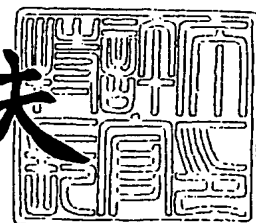


**PRIORITY  
DOCUMENT**  
SUBMITTED OR TRANSMITTED IN  
COMPLIANCE WITH RULE 17.1(a) OR (b)

2 0 0 4 年 5 月 1 4 日

特許庁長官  
Commissioner,  
Japan Patent Office

今 井 康 夫



【書類名】 特許願  
【整理番号】 J030129IK0  
【あて先】 特許庁長官殿  
【国際特許分類】 H03K 19/173  
G06F 7/00

## 【発明者】

【住所又は居所】 福岡県福岡市中央区今川 1 丁目 1 0 - 4 3 - 4 0 3

【氏名】 笹尾 勤

## 【発明者】

【住所又は居所】 神奈川県足柄下郡箱根町強羅 1 3 2 0 - 1 2 3 1

【氏名】 井口 幸洋

## 【特許出願人】

【識別番号】 502428467

【氏名又は名称】 笹尾 勤

## 【代理人】

【識別番号】 100099508

## 【弁理士】

【氏名又は名称】 加藤 久

【電話番号】 092-413-5378

## 【手数料の表示】

【予納台帳番号】 037590

【納付金額】 21,000円

## 【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 ルックアップテーブル・カスケード論理回路

【特許請求の範囲】

【請求項 1】 複数の入力変数を有する目的論理関数を関数分解して得られる複数の分解関数を、逐次計算することにより、目的論理関数の演算を行うルックアップテーブル・カスケード論理回路であって、

前記各分解関数をルックアップテーブルとして記憶する、直列に配列された複数の論理関数メモリと、

前段の前記論理関数メモリの複数の出力のうち、後段の前記論理関数メモリの複数の入力に接続されるものの選択及び接続関係に関する接続情報を記憶する接続メモリと、

前記接続メモリより出力される接続順序に従い、前段の前記論理関数メモリの出力と後段の前記論理関数メモリの入力とを選択的に接続する接続回路と、

最後段の前記論理関数メモリの出力を最前段の前記論理関数メモリに選択的に入力するフィードバック回路と、

を備えていることを特徴とするルックアップテーブル・カスケード論理回路。

【請求項 2】 前記フィードバック回路は、  
最後段の前記論理関数メモリの複数の出力のうち、最前段の前記論理関数メモリの複数の入力に接続されるものの選択及び接続関係に関する接続情報を記憶するフィードバック用接続メモリと、

最後段の前記論理関数メモリの出力を最前段の前記論理関数メモリの入力に選択的に接続するフィードバック用接続回路と、

を備えていることを特徴とする請求項 1 記載のルックアップテーブル・カスケード論理回路。

【請求項 3】 最後段の前記論理関数メモリの複数の出力を一時的に記憶し、前記フィードバック用接続回路に出力する中間変数バッファを備えていることを特徴とする請求項 2 記載のルックアップテーブル・カスケード論理回路。

【請求項 4】 前記各論理関数メモリ並びに前記各接続メモリ及び前記フィードバック用接続メモリについて、当該論理関数メモリ又は接続メモリ若しくは

フィードバック用接続メモリ内の入出力が可能なページを選択的に切り換えるページ切換手段を備えたことを特徴とする請求項1乃至3の何れかに記載のルックアップテーブル・カスケード論理回路。

【請求項5】 前記接続回路又はフィードバック用接続回路は、クロスバスイッチ、マルチプレクサ、若しくはシフタ回路により構成されていることを特徴とする請求項1乃至4の何れかに記載のルックアップテーブル・カスケード論理回路。

【請求項6】 前記接続回路又はフィードバック用接続回路は、前段の前記論理関数メモリ又は最後段の前記論理関数メモリの出力を、論理関数の演算結果を出力する出力ラインに接続することが可能であることを特徴とする請求項1乃至3の何れかに記載のルックアップテーブル・カスケード論理回路。

【請求項7】 前記各接続回路又はフィードバック用接続回路の前記出力ラインからの出力のうちの何れかを選択し出力する出力段選択回路を備えていることを特徴とする請求項6記載のルックアップテーブル・カスケード論理回路。

【請求項8】 前記出力段選択回路が選択した出力を、  
w入力1出力 ( $w \geq 2$ ) のマルチプレクサが多段にカスケード接続され、各段において、出力を取り出すことを可能とした回路を通して出力する出力ライン選択回路を備えていること

を特徴とする請求項7記載のルックアップテーブル・カスケード論理回路。

【請求項9】 前記接続メモリは、前段の前記論理関数メモリの複数個の出力及び前記入力変数の入力のうち、後段の前記論理関数メモリの複数個の入力に接続されるものの選択及び接続関係に関する接続情報を記憶するものであり、

前記接続回路は、前記接続メモリより出力される接続順序に従い、前段の前記論理関数メモリの出力と後段の前記論理関数メモリの入力とを選択的に接続するものであり、

前記フィードバック用接続メモリは、最後段の前記論理関数メモリの複数個の出力及び前記入力変数の入力のうち、最前段の前記論理関数メモリの複数個の入力に接続されるものの選択及び接続関係に関する接続情報を記憶するものであり、

前記フィードバック用接続回路は、前記フィードバック用接続メモリより出力される接続順序に従い、最後段の前記論理関数メモリの出力と最前段の前記論理関数メモリの入力とを選択的に接続するものであることを特徴とする請求項 2 乃至 8 の何れか一記載のルックアップテーブル・カスケード論理回路。

【請求項 10】 前記各論理関数メモリにおいては、その入力のうちの少なくとも 1 つには、前記接続回路を介することなく前記入力変数を直接入力することを特徴とする請求項 9 記載のルックアップテーブル・カスケード論理回路。

【請求項 11】 前記各入力変数を記憶する入力変数レジスタと、  
前記入力変数レジスタから出力される入力変数のうちの一部を選択し、前記論理関数メモリ又は前記接続回路若しくは前記フィードバック用接続回路に入力する入力選択回路と、  
を備えていることを特徴とする請求項 1 乃至 10 の何れか一に記載のルックアップテーブル・カスケード論理回路。

【請求項 12】 前記入力選択回路は、シフタ回路により構成されていることを特徴とする請求項 11 に記載のルックアップテーブル・カスケード論理回路。

【請求項 13】 前記入力選択回路は、前記各接続回路若しくはフィードバック用接続回路又は前記各論理関数メモリのそれぞれに対して設けられていることを特徴とする請求項 11 又は 12 に記載のルックアップテーブル・カスケード論理回路。

【請求項 14】 前記各論理関数メモリは、前段又は最後段の前記論理関数メモリの出力の一部が、前記接続回路又はフィードバック用接続回路を介することなく後段の前記論理関数メモリの入力の一部に接続されていることを特徴とする請求項 1 乃至 13 の何れか一に記載のルックアップテーブル・カスケード論理回路。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、プログラム可能な論理回路に係り、特に、目的論理関数を関数分解して得られる分解関数を、逐次計算することにより、目的論理関数の演算を行うルックアップテーブル・カスケード論理回路に関する。

#### 【0002】

##### 【従来の技術】

近年、論理回路の設計においては、マトリックス状に配列された複数の論理セルの内容と論理セル間の配線の接続とをメモリの書き換えによって変更することが可能なフィールド・プログラマブル・ゲートアレイ (Field Programmable Gate Array。以下、「FPGA」という。) が広く用いられるようになっている。FPGA は、プログラムに従ってソフトウェア的に演算を行うマイクロ・プロセッサ (以下、「MPU」という。) とは異なり、ハードウェア的に演算を行うことから、論理関数の演算実行速度が速いという特徴を有する。

#### 【0003】

その一方、FPGAは、物理的な配線の引き回しをプログラムにより変更することから、配線遅延を考慮した設計が必要とされる。そのため、論理回路の設計に長時間を要する。また、各論理セル間の配線の引き回し方により配線遅延時間が変わる。そのため、設計時において論理回路の演算時間の予測が困難である。

#### 【0004】

そこで、上記FPGAの欠点をカバーするプログラム可能な論理回路として、ルックアップテーブル・カスケード論理回路が提案されている (非特許文献1, 2 参照)。以下、ルックアップテーブル・カスケード論理回路について簡単に説明する。

#### 【0005】

図24はルックアップテーブル・カスケード論理回路の原理を表す図である。

ルックアップテーブル・カスケード論理回路において論理関数の演算を行う場合、まず、演算を行う論理関数 (目的論理関数)  $f(X)$  を  $p$  個の分解関数  $\{f_i(X_i) ; i=0, \dots, p-1\}$  に関数分解する。ここで、 $X=(x_0, \dots, x_{n-1})$  は入力変数を表す。また、入力変数の集合を  $|X|$  で表すと、 $|X|=|X_0| \cup \dots \cup |X_{p-1}|$ ,  $|X_i| \cap |X_j| = \emptyset$  ( $i \neq j ; i, j \in \{0, \dots, p-1\}$ ) である。各分解関数  $f_i$  の出力変数 (一般に、ベクト

ル) を  $Y_{i+1}$  で表す。以下、 $X, Y$  の変数の個数を  $|X|, |Y|$  のように表すこととし、特に、 $|X|=n$ 、 $|X_i|=n_i$ 、 $|Y_i|=u_i$  と表記する。

#### 【0006】

任意の目的論理関数  $f$  は、 $n_0=s$  ( $s$  は分解関数  $f_0$  の入力変数の数)、 $n_i=s-u_i$  ( $0 < i < p-1$ )、 $n_{p-1}=s-u_{p-1}-t$  ( $0 \leq t \leq s-2$ ) となるように関数分解することが可能である。そこで、目的論理関数  $f$  を  $p-1$  個の  $s$  入力の分解関数  $f_j$  ( $j \in \{0, \dots, p-2\}$ ) と 1 個の  $s-t$  入力の分解関数  $f_{p-1}$  とに分解する。そして、各分解関数  $f_k$  ( $k \in \{0, \dots, p-1\}$ ) を真理値表により表し、これをルックアップテーブルとする。

#### 【0007】

図 24 において、 $p$  個のルックアップテーブル  $LUT_0 \sim LUT_{p-1}$  は、それぞれ、分解関数  $\{f_i; i=0, \dots, p-1\}$  を真理値表により表現したものである。それぞれのルックアップテーブル  $LUT_k$  は、 $s$  入力  $u_{k+1}$  出力の論理関数メモリを用いて実現することができる。このような論理関数メモリを、図 24 に示すようにカスケード接続することにより、ルックアップテーブル・カスケード論理回路が実現される。

#### 【0008】

ルックアップテーブル・カスケード論理回路では、目的論理関数  $f$  の入出力変数の個数が多い場合には、多数のルックアップテーブルをカスケード接続する必要があるため、最適化された FPGA に比べれば演算速度は遅くなる場合もある。しかし、ルックアップテーブル・カスケード論理回路では、各分解関数の演算について、論理関数メモリを用いてハードウェア的に行うことから、MPU に比べると演算速度を高速化することができる。そして、ルックアップテーブルの段数によって、一義的に演算速度が定まることから、論理回路の設計時に目的論理関数の演算時間を予測することが容易である。また、互いに隣接する論理関数メモリ間でのみ配線を行えばよいことから、論理回路の設計時には配線遅延の影響を考慮する必要がなく、FPGA に比べると論理回路設計が格段に容易になる。

#### 【0009】

##### 【非特許文献 1】

笹尾勤，松浦宗寛，井口幸洋，“多出力関数のカスケード実現と再構

成可能ハードウェアによる実現”，電子情報通信学会FTS研究会，FTS2001-8，pp. 57-64，三重大学(2001-04)．

#### 【非特許文献 2】

T. Sasao, M. Matsuura, and Y. Iguchi, "A cascade realization of multiple-output function for reconfigurable hardware," International Workshop on Logic and Synthesis (IWLS01), Lake Tahoe, CA, June 12-15, 2001. pp.225-230.

#### 【0010】

##### 【発明が解決しようとする課題】

しかしながら、実際に図 24 に示したようなルックアップテーブル・カスケード論理回路により目的論理関数の演算を行う場合には、目的論理関数を関数分解して得られる分解関数のルックアップテーブル間のレイル数（二つのルックアップテーブル間の線数（中間変数の数））や各ルックアップテーブルの入力変数の個数が目的論理関数により異なる。そのため、種々の目的論理関数をルックアップテーブル・カスケード論理回路で実現しようとした場合、入力変数の入力ラインの数及びレイル数が異なる種々のルックアップテーブル・カスケードが必要となり、実用性に欠ける。

#### 【0011】

そこで、多様な目的論理関数に対応可能なルックアップテーブル・カスケード論理回路とするために、各論理関数メモリの入力ビット数に余裕をもたせ、あらかじめビット数を大きめにとっておくことが考えられる。しかし、論理関数メモリのメモリ容量は、論理関数メモリの入力ビット数の増加に伴い増大する。そして、使用されないメモリ領域が増大し、無駄が多くなる。また、メモリ容量の増大に伴いメモリのレイアウト面積が増大するため、回路の高集積化が困難となり、回路消費電力も大きくなる。

#### 【0012】

また、目的論理関数の入力変数が多くなった場合、多数の分解関数に関数分解してルックアップテーブル・カスケードを構成する必要が生じる。しかし、分解関数の数が増加すると、多数の論理関数メモリを直接接続する必要が生じ、回路



面積が大きくなる。従って、ルックアップテーブル・カスケード論理回路の小型化・高集積化が困難となる。

#### 【0013】

そこで、本発明の目的は、目的論理関数に応じて、各論理関数メモリ間の入力変数の入力ライン数とレイル数とを柔軟に変化させることが可能で、論理関数メモリの入力数、延いては論理関数メモリのメモリ容量を必要最小限に抑えて設計することが可能であり、また、分解関数の数に応じてルックアップテーブル・カスケードの段数を柔軟に変化させることが可能なルックアップテーブル・カスケード論理回路を提供することにある。

#### 【0014】

##### 【課題を解決するための手段】

本発明に係るルックアップテーブル・カスケード論理回路の第1の構成は、複数の入力変数を有する目的論理関数を関数分解して得られる複数の分解関数を、逐次計算することにより、目的論理関数の演算を行うルックアップテーブル・カスケード論理回路であって、前記各分解関数をルックアップテーブルとして記憶する、直列に配列された複数の論理関数メモリと、前段の前記論理関数メモリの複数個の出力のうち、後段の前記論理関数メモリの複数個の入力に接続されるものの選択及び接続関係に関する接続情報を記憶する接続メモリと、前記接続メモリより出力される接続順序に従い、前段の前記論理関数メモリの出力と後段の前記論理関数メモリの入力とを選択的に接続する接続回路と、最後段の前記論理関数メモリの出力を最前段の前記論理関数メモリに選択的に入力するフィードバック回路とを備えていることを特徴とする。

#### 【0015】

この構成によれば、目的関数の分解関数の真理値をルックアップテーブルとして論理関数メモリに記憶させておき、目的関数に応じて、各論理関数メモリの入出力の選択及び接続関係を接続メモリに記憶させておく。そして、接続メモリに記憶された選択及び接続関係に従って、接続回路によって前段の論理関数メモリと後段の論理関数メモリの入出力が接続される。これにより、目的関数に適合させて接続関係を自由に変更することができるため、論理回路をプログラマブルに

構成することが可能となる。

#### 【0016】

また、フィードバック回路によって、最後段の論理関数メモリの出力は、最前段の論理関数メモリに選択的に入力することが可能である。従って、ルックアップテーブル・カスケード論理回路によって演算することが可能な論理関数の大きさの自由度が大きくなる。すなわち、目的関数の入力変数の数が各論理関数メモリの入力端子数の和よりも多い場合であっても、最後段の論理関数メモリの出力を最前段の論理関数メモリにフィードバックさせて、同じ論理関数メモリを重複して使用して分解関数の演算を行わせることができる。従って、分解関数の数に応じてルックアップテーブル・カスケードの段数を柔軟に変化させることが可能となる。そのため、設計自由度が大きくなる。

#### 【0017】

更に、フィードバック回路を備えたことで、ルックアップテーブル・カスケードの段数を多くすることができるので、各段のルックアップテーブルの大きさは小さくすることができる。従って、各段の論理関数メモリは、メモリ容量の比較的小さいものを使用することができる。そのため、各論理関数メモリの消費電力は低く抑えることが可能である。更に、演算の流れに従って、一部の論理関数メモリのみがデータ読み出しの動作をする。そして、この動作を行うメモリのみが主として電力を消費する。従って、MPUやFPGAに比べ、低消費電力で動作させることが可能となる。また、データ読み出しの動作をしていないメモリは、低電力モードにするように構成すれば、更に消費電力の低減が可能となる。

#### 【0018】

本発明に係るルックアップテーブル・カスケード論理回路の第2の構成は、前記第1の構成において、前記フィードバック回路は、最後段の前記論理関数メモリの複数個の出力のうち、最前段の前記論理関数メモリの複数個の入力に接続されるものの選択及び接続関係に関する接続情報を記憶するフィードバック用接続メモリと、最後段の前記論理関数メモリの出力を最前段の前記論理関数メモリの入力に選択的に接続するフィードバック用接続回路と、を備えていることを特徴とする。

## 【0019】

この構成によれば、最後段の論理関数メモリの出力を最前段の論理関数メモリの入力にフィードバックさせる際に、最後段の論理関数メモリから出力される中間変数の選択及び最前段の論理関数メモリとの接続関係を、フィードバック用接続回路とフィードバック用接続メモリによってプログラマブルに設定することができる。従って、フィードバック後に最前段の論理関数メモリで演算を行う分解関数のルックアップテーブルの構成の自由度が高まる。また、フィードバックの際の最前段の論理関数メモリの入出力端子を有効に利用することが可能となるとともに、最前段の論理関数メモリのメモリ容量を効率よく活用することが可能となる。

## 【0020】

本発明に係るルックアップテーブル・カスケード論理回路の第3の構成は、前記第2の構成において、最後段の前記論理関数メモリの複数個の出力を一時的に記憶し、前記フィードバック用接続回路に出力する中間変数バッファを備えていることを特徴とする。

## 【0021】

この構成により、各論理関数メモリの演算をパイプライン処理によって行った場合に、発振することを防止できる。すなわち、最前段の論理関数メモリから最後段の論理関数メモリへ、順次非同期に演算処理を1ループだけ行った後、最後段の論理関数メモリから出力される中間変数の値を、一旦中間変数バッファに記憶する。そして、中間変数バッファの値が確定した時点で、次のループの演算処理を開始させる。これにより、各論理関数メモリに記憶された分解関数の形によらず、各論理関数メモリの出力が発振により不安定化することが防止される。

## 【0022】

本発明に係るルックアップテーブル・カスケード論理回路の第4の構成は、前記第1乃至3の何れかの構成において、前記各論理関数メモリ並びに前記各接続メモリ及び前記フィードバック用接続メモリについて、当該論理関数メモリ又は接続メモリ若しくはフィードバック用接続メモリ内の入出力が可能なページを選択的に切り換えるページ切換手段を備えたことを特徴とする。

## 【0023】

この構成により、ページ切換手段によって論理関数メモリ又は接続メモリ若しくはフィードバック用接続メモリ内の入出力が可能なページを選択的に切り換えることで、1つの論理関数メモリで複数の種類の分解関数の演算を行うことが可能となる。

## 【0024】

本発明に係るルックアップテーブル・カスケード論理回路の第5の構成は、前記第1乃至4の何れか一の構成において、前記接続回路又はフィードバック用接続回路は、クロスバススイッチ、マルチプレクサ、若しくはシフタ回路により構成されていることを特徴とする。

## 【0025】

このように、接続回路をクロスバススイッチ、マルチプレクサ、若しくはシフタ回路とすることで、簡単な回路により接続回路を実現することができる。

## 【0026】

特に、回路の演算速度の高速化が要求される場合には、接続回路としてクロスバススイッチを使用することが好ましい。1本の配線上の切換スイッチの個数が少なく、切換スイッチにおける遅延を極力減らすことが可能だからである。

## 【0027】

また、回路の特に小型化が要求される場合には、接続回路としてシフタ回路を使用することが好ましい。接続順序の自由度は少ないが、接続メモリに必要とされる、接続回路の接続切換を制御するための制御線の数をも最小限とすることが可能となり、回路のレイアウト面積及び消費電力を小さくできるからである。

## 【0028】

本発明に係るルックアップテーブル・カスケード論理回路の第6の構成は、前記第1乃至3の何れか一の構成において、前記接続回路又はフィードバック用接続回路は、前段の前記論理関数メモリ又は最後段の前記論理関数メモリの出力を、論理関数の演算結果を出力する出力ラインに接続することが可能であることを特徴とする。

## 【0029】

この構成により、カスケード接続された論理関数メモリうちの任意のものから、その出力を取り出すことができる。従って、目的論理関数の分解関数の段数が少ない場合には、カスケードの途中の論理関数メモリの出力を出力変数として取り出すことで、演算処理を早期に終了させて演算速度を高速化することができる。

#### 【0030】

本発明に係るルックアップテーブル・カスケード論理回路の第7の構成は、前記第6の構成において、前記各接続回路又はフィードバック用接続回路の前記出力ラインからの出力のうちの何れかを選択し出力する出力段選択回路を備えていることを特徴とする。

#### 【0031】

この構成により、出力回路により、各接続回路の前記出力ラインからの出力又はフィードバック用接続回路の前記出力ラインからの出力のうち、目的論理関数の演算結果が出力されるものを選択して出力させることができる。

#### 【0032】

本発明に係るルックアップテーブル・カスケード論理回路の第8の構成は、前記第7の構成において、前記出力段選択回路が選択した出力を、 $w$ 入力1出力（ $w \geq 2$ ）のマルチプレクサが多段にカスケード接続され、各段において、出力を取り出すことを可能とした回路を通して出力する出力ライン選択回路を備えていることを特徴とする。

#### 【0033】

この構成により、出力段選択回路により選択された段の分解関数の出力変数を出力ライン選択回路によって選択し、出力変数の個数を $1/2$ 以下にして出力することが可能となる。すなわち、出力段選択回路により選択された段の分解関数の出力変数の個数が、論理関数メモリの出力可能なラインの数の $1/2$ 以下の場合、入力変数のうちの一部のものの値に基づいて、出力ライン選択回路により、更に分解関数の出力変数を選択することができる。すなわち、出力ライン選択回路において更に論理演算を行うことができる。従って、各論理関数メモリに入力する入力変数の個数を削減することができるため、論理関数メモリのメモリ使用

効率が向上する。また、ルックアップテーブル・カスケード論理回路全体で演算が可能な論理関数の入力変数の個数に関する制限数を広げることが可能となる。

#### 【0034】

本発明に係るルックアップテーブル・カスケード論理回路の第9の構成は、前記第2乃至8の何れか一の構成において、前記接続メモリは、前段の前記論理関数メモリの複数個の出力及び前記入力変数の入力のうち、後段の前記論理関数メモリの複数個の入力に接続されるものの選択及び接続関係に関する接続情報を記憶するものであり、前記接続回路は、前記接続メモリより出力される接続順序に従い、前段の前記論理関数メモリの出力と後段の前記論理関数メモリの入力とを選択的に接続するものであり、前記フィードバック用接続メモリは、最後段の前記論理関数メモリの複数個の出力及び前記入力変数の入力のうち、最前段の前記論理関数メモリの複数個の入力に接続されるものの選択及び接続関係に関する接続情報を記憶するものであり、前記フィードバック用接続回路は、前記フィードバック用接続メモリより出力される接続順序に従い、最後段の前記論理関数メモリの出力と最前段の前記論理関数メモリの入力とを選択的に接続するものであることを特徴とする。

#### 【0035】

この構成により、前段又は最終段の論理関数メモリから出力される中間変数の個数と、各段の論理関数メモリに入力する入力変数の個数とを、接続メモリ及び接続回路又はフィードバック用接続メモリ及びフィードバック用接続回路を用いて、各分解関数に適合させて変更することが可能となる。すなわち、各段の分解関数のルックアップテーブル間のレイル数（二つのルックアップテーブル間の線数（中間変数の数））が少ないときには、その分だけ入力変数の個数を増やすことができ、逆に各段の分解関数のルックアップテーブル間のレイル数が多いときには入力変数の個数を減らすことができる。従って、各論理関数メモリの限られた数の入力を有効に活用することができ、延いては、各論理関数メモリのメモリ容量を有効に活用することができる。

#### 【0036】

本発明に係るルックアップテーブル・カスケード論理回路の第10の構成は、

前記第 9 の構成において、前記各論理関数メモリにおいては、その入力の中の少なくとも 1 つには、前記接続回路を介することなく前記入力変数を直接入力することを特徴とする。

#### 【0037】

最初の段の論理関数メモリについては、総ての入力に対して入力変数が入力される。一方、最初の段以外の各論理関数メモリにおいては、その入力の一部には前段の論理関数メモリの出力が入力され、他の入力には、入力変数が入力される。しかしながら、最初の段以外の各論理関数メモリには、最低限 1 つの入力変数が入力される。従って、最初の段以外の各論理関数メモリの入力の中の少なくとも 1 つに、接続回路を介することなく入力変数を直接入力することにより、接続回路の入力ラインの数を減らすことができる。これにより、接続回路及び接続メモリの出力の配線数を減らすことができるため、回路を小型化することが可能となる。

#### 【0038】

本発明に係るルックアップテーブル・カスケード論理回路の第 11 の構成は、前記第 1 乃至 10 の何れか一の構成において、前記各入力変数を記憶する入力変数レジスタと、前記入力変数レジスタから出力される入力変数のうちの一部を選択し、前記論理関数メモリ又は前記接続回路若しくは前記フィードバック用接続回路に入力する入力選択回路と、を備えていることを特徴とする。

#### 【0039】

この構成によれば、入力選択回路は、入力変数レジスタに記憶された入力変数のうち、少なくとも実行中の分解関数の演算で使用される入力変数を含む変数を選択して出力する。従って、論理関数メモリ又は接続回路若しくはフィードバック用接続回路に入力される入力変数の個数を減らすことができる。そのため、接続回路及び接続メモリ並びにフィードバック用接続回路及びフィードバック用接続メモリの出力の配線数を減らすことができるため、回路を小型化することが可能となる。

#### 【0040】

本発明に係るルックアップテーブル・カスケード論理回路の第 12 の構成は、

前記第 11 の構成において、前記入力選択回路は、シフタ回路により構成されていることを特徴とする。

#### 【0041】

この構成により、入力変数の選択は、入力選択回路であるシフタ回路と接続回路とにより行われる。すなわち、シフタ回路によって入力レジスタに記憶された入力変数のうち、先頭から  $k$  ビットだけシフトさせた入力変数から所定のビット分の入力変数が選択されて接続回路又は論理関数メモリに出力される。接続回路は、入力された入力変数のうち、必要なものを選択して論理関数メモリに入力する。従って、簡易な回路によって入力選択回路を構成することができる。また、入力選択回路における入力変数の選択も、入力変数のシフトビット数を指定するだけでよいため、入力選択回路の制御ラインが少なく、配線面積を減らすことができる。

#### 【0042】

本発明に係るルックアップテーブル・カスケード論理回路の第 13 の構成は、前記第 11 又は 12 の構成において、前記入力選択回路は、前記各接続回路若しくはフィードバック用接続回路又は前記各論理関数メモリのそれぞれに対して設けられていることを特徴とする。

#### 【0043】

この構成により、各接続回路及び各論理関数メモリに対して入力する入力変数を、各入力選択回路により個別に選択することができる。従って、各論理関数メモリの演算を同時に行うことが可能であり、各論理関数メモリで並行して演算処理を行うことが可能となる。

#### 【0044】

本発明に係るルックアップテーブル・カスケード論理回路の第 14 の構成は、前記第 1 乃至 13 の何れか一の構成において、前記各論理関数メモリは、前段又は最後段の前記論理関数メモリの出力の一部が、前記接続回路又はフィードバック用接続回路を介すことなく後段の前記論理関数メモリの入力の一部に接続されていることを特徴とする。

#### 【0045】



最初の段以外の各論理関数メモリにおいては、その入力の一部には前段の論理関数メモリの出力が入力され、他の入力には、入力変数が入力される。しかしながら、最初の段以外の各論理関数メモリには、最低1つは前段の論理関数メモリの出力が入力される。従って、最初の段以外の各論理関数メモリの入力のうちの一部を、接続回路を介することなく前段の論理関数メモリの出力を直接入力することにより、接続回路の入力ラインの数を減らすことができる。これにより、接続回路及び接続メモリの素子数及び出力の配線数を減らすことができるため、回路を小型化することが可能となる。

【0046】

【発明の実施の形態】

以下、本発明の一実施形態について、図面を参照しながら説明する。

【0047】

(実施形態1)

図1は本発明の実施形態1に係るルックアップテーブル・カスケード論理回路の全体構成を表すブロック図である。

【0048】

本発明の実施形態1に係るルックアップテーブル・カスケード論理回路は、入力変数レジスタ1、入力変数選択回路2-0～2-3、入力選択メモリ3-0～3-3、論理関数メモリ4-0～4-3、接続回路5-0～5-3、接続メモリ6-0～6-3、ページ選択メモリ8-0～8-3、及び演算制御部10を備えている。尚、本実施形態においては、各回路は共通のクロックによって同期して動作するものとする。

【0049】

入力変数レジスタ1は、ルックアップテーブル・カスケード論理回路によって演算を行う論理関数（目的論理関数） $f(X)$ の演算に用いられる $n$ 個の入力変数 $X = (x_1, \dots, x_n)$ （但し、 $n \in \text{自然数}$ ）を記憶する。論理関数メモリ4-0～4-3は、目的論理関数 $f$ を関数分解して得られる分解関数 $\{f_i; i \in \{0, 1, 2, 3\}\}$ の真理値表を、ルックアップテーブルとして記憶する。尚、本実施形態においては、一例として、論理関数メモリ4- $i$ （ $i = 0, 1, 2, 3$ ）の段数を4段としてい

るが、一般にはこの段数は任意に設定することができる。これらの論理関数メモリ 4-0～4-3 は、直列に配列され、それぞれ接続回路 5-1～5-3 を介して直列に接続されている。

#### 【0050】

また、論理関数メモリ 4-3 の出力側は、フィードバック用の接続回路 5-0 を介して論理関数メモリ 4-0 の入力側に接続されている。

#### 【0051】

入力変数選択回路 2-0～2-3 は、入力変数レジスタ 1 から出力される  $n$  個の入力変数  $X=(x_1, \dots, x_n)$  のうちから、それぞれ、各段の論理関数メモリ 4-0～4-3 に記憶された分解関数  $\{f_i; i \in \{0, 1, 2, 3\}\}$  の真理値表の入力変数  $X_i$  ( $i \in \{0, 1, 2, 3\}$ ) を選択し、接続回路 5-0～5-3 へ出力する。入力選択メモリ 3-0～3-3 は、それぞれ、入力選択回路 2-0～2-3 の入力変数の選択に関する情報（以下、「入力選択情報」という。）が記憶されている。入力変数選択回路 2-0～2-3 は、各入力選択メモリ 3-0～3-3 から出力される入力選択信号に基づいて、入力変数の選択の切り換えを行う。

#### 【0052】

各接続回路 5- $i$  ( $i \in \{1, 2, 3\}$ ) は、論理関数メモリ 4- $(i-1)$  及び入力選択回路 2- $i$  より入力される、中間変数  $Y_i$  及び入力変数  $X_i$  を、順序を変えて、後段の論理関数メモリ 4- $i$  及び外部出力ライン 7- $i$  へ接続する。また、接続回路 5-0 は、論理関数メモリ 4-3 及び入力選択回路 2-0 より入力される、中間変数  $Y_i$  及び入力変数  $X_i$  を、順序を変えて、後段の論理関数メモリ 4-0 及び外部出力ライン 7-0 へ接続する。各接続メモリ 6- $i$  ( $i \in \{0, 1, 2, 3\}$ ) は、各接続回路 5- $i$  の接続関係に関する情報（以下、「接続情報」という。）を記憶している。接続回路 5- $i$  は、接続メモリ 6- $i$  から出力される接続情報信号に基づき、接続関係の切り換えを行う。

#### 【0053】

各ページ選択メモリ 8- $i$  ( $i \in \{0, 1, 2, 3\}$ ) は、目的関数の演算段数に対応して、論理関数メモリ 4- $i$  に記憶されたルックアップテーブルのうちの使用するものが格納されているページ番号（以下、「ページ情報」という。）が記憶さ

れている。論理関数メモリ 4-i は、ページ選択メモリ 8-i が出力するページ番号に従ってページを設定する。

#### 【0054】

演算制御部 10 は、ルックアップテーブル・カスケード論理回路全体の演算処理の制御を行う。

#### 【0055】

図 2 は図 1 の入力変数選択回路 2-i ( $i \in \{0, 1, 2, 3\}$ ) の構成を表す回路図である。尚、この回路図は、動作原理を説明するために、簡略化して記載している。

#### 【0056】

本実施形態の入力変数選択回路 2-i ( $i \in \{0, 1, 2, 3\}$ ) は、図 2 のようなシフト回路により構成されている。尚、図 2 では、説明の都合上、17 入力 8 出力の入力変数選択回路 2-i の例を示しているが、入出力の数は、これに限られるものではない。

#### 【0057】

入力変数選択回路 2-i ( $i \in \{0, 1, 2, 3\}$ ) の入力端子 in(00)~in(16) は、入力変数レジスタ 1 の出力端子に接続されている。従って、入力端子 in(00)~in(16) からは入力変数  $X = (x_1, \dots, x_n)$  が入力される。また、入力変数選択回路 2-i ( $i \in \{0, 1, 2, 3\}$ ) の出力端子 out(00)~out(07) は、接続回路 5-i の入力側端子の一部に接続されている。

#### 【0058】

入力変数選択回路 2-i ( $i \in \{0, 1, 2, 3\}$ ) は、入力側から、8 ビットシフト回路 11-3、4 ビットシフト回路 11-2、2 ビットシフト回路 11-1、及び 1 ビットシフト回路 11-0 が直列に接続されている。これにより、入力変数選択回路 2-i ( $i \in \{0, 1, 2, 3\}$ ) は、入力端子 in(00)~in(16) の内容を 0~16 ビットの範囲でシフトさせて、連続する 8 ビット分を出力端子 out(00)~out(07) に出力させることができる。

#### 【0059】

各シフト回路 11-0~11-3 には、それぞれ 1 ビットの制御ライン shf0~sh

f3が接続されている。制御ラインshf j ( $j \in \{0, 1, 2, 3\}$ ) が0のときは、シフト回路11-iは接続のシフトを行わず、制御ラインshf j ( $j \in \{0, 1, 2, 3\}$ ) が1のときは、シフト回路11-iは接続のシフトを行う。尚、入力変数選択回路2-i ( $i \in \{0, 1, 2, 3\}$ ) の各制御ラインshf0~shf3は、入力選択メモリ3-iの出力に接続されている。すなわち、入力選択メモリ3-0から読み出された4ビットのメモリの内容(すなわち、入力選択情報)が、選択制御信号としてそのまま制御ラインshf0~shf3に出力され、入力選択回路2-i ( $i \in \{0, 1, 2, 3\}$ ) における入力変数のシフト量が設定される。

#### 【0060】

尚、図2では、簡略化のために省略しているが、シフタ回路11-0~11-3内の各パストランジスタにより入力変数Xの信号振幅が減衰する。従って、入力変数Xの信号振幅を補償するために、シフタ回路の数段ごとに増幅器(バッファ)を挿入しておくことが必要となる。

#### 【0061】

図3は図1の論理関数メモリ4-i ( $i \in \{0, 1, 2, 3\}$ 、以下、符号4-0~4-3をまとめていう場合には、符号4で表す。)の構成を表すブロック図である。

#### 【0062】

論理関数メモリ4は、その内部に、mページからなるメモリアレイFM<sub>0</sub>~FM<sub>m-1</sub>を備えている。これらのメモリアレイには、分解関数 $f_i(j)$  ( $i \in \{0, 1, 2, 3\}$ ,  $j \in \{0, \dots, m-1\}$ ) の真理値表がルックアップテーブルLUT(j)として格納されている。

#### 【0063】

尚、ここで、添字「(j)」 ( $j \in \{0, \dots, m-1\}$ ) は、論理関数メモリ4のjページ目を表す。本実施形態におけるルックアップテーブル・カスケード論理回路では、複数の分解関数 $\{f_i(j)\}$ を、それぞれの論理関数メモリ4の第jページに記憶させておき、目的論理関数に応じて、又はループ回数に応じて、ページを切り換えて、目的論理関数を選択することが可能とされている。

#### 【0064】

また、論理関数メモリ4-i ( $i \in \{0, 1, 2, 3\}$ ) は、ページデコーダ15及び

アドレスデコーダ 16 を備えている。ページデコーダ 15 は、ページ選択メモリ 8-i から入力されるページ選択番号  $p$  ( $p_k$ ) に基づいて、メモリアレイ  $FM_k$  のメモリアクセスを可能とする。アドレスデコーダ 16 は、接続回路から  $N_{in}$  ビットの入力ライン 17 を介して入力される変数  $(X_i, Y_i)$  ( $i \in \{0, 1, 2, 3\}$ . 但し、ルックアップテーブル・カスケードの 1 段目の場合は、 $Y_i = \phi$ 。) に基づき、メモリアレイ  $FM_k$  内のメモリ内容を読み出すアドレスを選択する。この選択されたアドレスのメモリセルには、分解関数  $f_k(j)$  の真理値  $f_k(j)(X_k, Y_k)$  が格納されている。メモリアレイ  $FM_k$  は、ページデコーダ 15 及びアドレスデコーダ 16 により、メモリセル内のアドレスが指定されると、当該メモリセル内に記憶されているデータを、中間変数  $Y_{k+1} = (y_{k+1,1}, \dots, y_{k+1,|Y_{k+1}|})$  として出力ライン 18 に出力する。

#### 【0065】

図 4 は図 1 の接続回路 5-i ( $i \in \{0, 1, 2, 3\}$ . 以下、符号 5-0 ~ 5-3 をまとめていう場合には、符号 5 で表す。) の周辺の回路ブロック図である。尚、この回路図は、動作原理を説明するために、簡略化して記載している。

#### 【0066】

本実施形態における接続回路 5 は、入力ライン  $i_0 \sim i_{15}$  からの入力信号を循環的に任意のビット数だけシフトさせて出力ライン  $o_1 \sim o_{15}$  に出力するシフト回路により構成されている。尚、図 4 においては、説明の都合上、一例として、16 ビット入力 16 ビット出力の接続回路を記載しているが、入出力のビット数は、特にこれに限定されない。

#### 【0067】

接続回路 5 は、入力ライン  $i_0 \sim i_{15}$  の側から、8 ビットシフト回路 20-3、4 ビットシフト回路 20-2、2 ビットシフト回路 20-1、1 ビットシフト回路 20-0 が直列に接続された構成とされている。各シフト回路 20-i ( $i \in \{0, 1, 2, 3\}$ ) のオン/オフは、接続メモリ 6 (符号 6-0 ~ 6-3 をまとめて符号 6 で表す。以下同じ。) の出力に接続された制御ライン  $s_j$  ( $j \in \{0, 1, 2, 3\}$ ) により制御される。

#### 【0068】

接続回路 5-k ( $k \in \{1, 2, 3\}$ ) の入力ライン  $i_0 \sim i_{15}$  のうち、 $i_0 \sim i_7$  は前段の論理関数メモリ 4- ( $k-1$ ) の出力側に接続されており、 $i_8 \sim i_{15}$  は入力変数選択回路 2-k の出力側に接続されている。また、接続回路 5-0 の入力ライン  $i_0 \sim i_{15}$  のうち、 $i_0 \sim i_7$  は前段の論理関数メモリ 4-3 の出力側に接続されており、 $i_8 \sim i_{15}$  は入力変数選択回路 2-0 の出力側に接続されている。

**【0069】**

一方、接続回路 5-k ( $k \in \{0, 1, 2, 3\}$ ) の出力ライン  $o_0 \sim o_{15}$  のうち、 $o_0 \sim o_7$  は、後段の論理関数メモリ 4-k の入力側に接続されており、 $o_8 \sim o_{15}$  は外部出力ライン 7-k となっている。

**【0070】**

各シフト回路 20-0 ~ 20-3 の接続関係のシフトがないときには、入力ライン  $i_0 \sim i_{15}$  は、それぞれ出力ライン  $o_0 \sim o_{15}$  に接続される。これにより、前段の論理関数メモリ 4- ( $k-1$ ) の出力である中間変数  $Y_k$  は、総て後段の論理関数メモリ 4-k に入力される。

**【0071】**

一方、シフト回路 20-3 のみが接続関係を 8 ビットシフトさせた場合には、入力ライン  $i_0 \sim i_7$  が、それぞれ、出力ライン  $o_8 \sim o_{15}$  に接続され、入力ライン  $i_8 \sim i_{15}$  が、それぞれ、出力ライン  $o_0 \sim o_7$  に接続される。これにより、前段の論理関数メモリ 4- ( $k-1$ ) の出力である中間変数  $Y_k$  は、総て外部出力ライン 7-k に出力される。また、入力変数選択回路 2-k から出力される入力変数  $X_k$  は、総て後段の論理関数メモリ 4-k に入力される。

**【0072】**

また、各シフト回路 20-0 ~ 20-2 により、接続関係を  $j$  ビット ( $1 \leq j \leq 7$ ) だけシフトさせた場合には、入力ライン  $i_{8+j} \sim i_{15}$ ,  $i_0 \sim i_{j-1}$  が、それぞれ、出力ライン  $o_8 \sim o_{15}$  に接続され、入力ライン  $i_j \sim i_{7+j}$  が、それぞれ、出力ライン  $o_0 \sim o_7$  に接続される。これにより、入力ライン  $i_8 \sim i_{7+j}$  に入力される  $j$  ビットの入力変数  $X_k$  が、後段の論理関数メモリ 4-k に入力されるとともに、入力ライン  $i_j \sim i_7$  に入力される  $8-j$  ビットの中間変数  $Y_k$  が後段の論理関数メモリ 4-k に入力される。従って、後段の論理関数メモリ 4-k に入力する入力変数  $X_k$  の変数の個

数 $|X_k|=n_k$ と、後段の論理関数メモリ4-kに入力する中間変数 $Y_k$ の変数の個数 $|Y_k|=u_k$ とを、 $s=n_k+u_k=8$ の条件下で自由に変更することが可能となっている。

#### 【0073】

図5は図1の接続メモリ6の構成を表すブロック図である。

接続メモリ6-i ( $i \in \{0, 1, 2, 3\}$ ) の内部には、 $m$ 個のメモリアレイ $CM_0 \sim CM_{m-1}$ が備えられている。各メモリアレイ $CM_0 \sim CM_{m-1}$ には、接続回路5-iの各シフト回路20-( $N_c-1$ )  $\sim$  20-0 (ここで、 $N_c$ は接続回路におけるシフト回路の段数を表し、図4では $N_c=4$ である。)のオン/オフ情報を表す接続情報( $s_{N_c-1}^{(j)}$ ,  $\dots, s_0^{(j)}$ ) ( $j \in \{0, \dots, m-1\}$ ) が記憶されている。尚、ここで、添字「 $(j)$ 」 ( $j \in \{0, \dots, m-1\}$ ) は、 $j$  ページ目に格納された接続情報であることを表す。

#### 【0074】

これらのメモリアレイ $CM_0 \sim CM_{m-1}$ の $N_c$ ビットの出力ライン $s_0 \sim s_{N_c-1}$ は、接続回路5-iの各シフト回路20-0  $\sim$  20-( $N_c-1$ )の制御ラインとなっている。

#### 【0075】

また、各接続メモリ6の内部には、ページデコーダ21が設けられている。ページデコーダ21は、演算制御部10から入力されるページ選択番号 $p_k$ に従って、 $k$ 番目のメモリアレイ $CM_k$ を選択する。ページデコーダ21により選択されたメモリアレイ $CM_k$ は、 $N_c$ ビットの接続情報( $s_{N_c-1}^{(k)}$ ,  $\dots, s_0^{(k)}$ )からなる接続制御信号を接続回路5に出力する。

#### 【0076】

図6は図1の演算制御部10の構成を表すブロック図である。

演算制御部10は、演算ステップレジスタ31、ステップカウンタ32、ページカウンタ34、及び出力コントローラ35を備えた構成からなる。

#### 【0077】

演算ステップレジスタ31には、目的関数を関数分解した後の分解関数の数である演算ステップ数が格納される。ステップカウンタ32は、現在演算が行われている分解関数の段数をカウントする。このステップカウンタ32には、演算ステップレジスタ31に格納された値が設定され、演算処理が1段進行するのに応

じて、内部に格納された値を1ずつ減じていくダウンカウンタによって構成されている。ステップカウンタ32は、そのカウント値*i*が0となったときに、終了信号ENDを出力する。また、ステップカウンタ32は、外部からリセット信号resetが入力されたとき、カウント値を演算ステップレジスタ31に格納された値に設定する。

#### 【0078】

ページカウンタ34は、入力選択メモリ3、接続メモリ6、及びページ選択メモリ8が出力するデータが格納されている各メモリのページ番号をカウントし、そのカウント値*k*をカウント信号pkとして出力する。ページカウンタ34は、アップカウンタであり、出力コントローラ35から出力されるチップ・イネーブル信号CE<sub>3</sub>の立ち下がりを検出したときに、カウント値*k*を1だけ増加させる。尚、ページカウンタ34は、リセット信号resetが入力されたとき、又はステップカウンタ32から終了信号ENDが入力されたときに、カウント値*k*を0にリセットする。

#### 【0079】

出力コントローラ35は、各々の論理関数メモリ4-0～4-3に対して、チップ・イネーブル信号CE<sub>0</sub>～CE<sub>3</sub>、及びアドレス・ストロープ信号ADSP<sub>0</sub>～ADSP<sub>3</sub>を出力することで、論理関数メモリ4-0～4-3の出力制御を行う。

#### 【0080】

尚、チップ・イネーブル信号CE<sub>*i*</sub>は、*i*段目の論理関数メモリ4-*i*を活性化する信号であり、チップ・イネーブル信号CE<sub>*i*</sub>が“1”（真値）となった（アサートされた）ときに、論理関数メモリ4-*i*が外部からの入力信号の受け付け及びデータの出力を行うことが可能となる。アドレス・ストロープ信号ADSP<sub>*i*</sub>は、論理関数メモリ4-*i*に対して、入力されたアドレスのラッチを制御するための信号である。論理関数メモリ4-*i*は、クロックエッジでアドレス・ストロープ信号ADSP<sub>*i*</sub>が“1”（真値）となっている（アサートされている）ときに、入力ライン*i*(00)～*i*(07)に入力されているデータを、内部のアドレスレジスタに設定する。また、アドレス・ストロープ信号ADSP<sub>*i*</sub>が“1”から“0”（偽値）に遷移すると、内部のアドレスレジスタに設定されているアドレス値をラッチする。尚



、チップ・イネーブル信号 $CE_i$ が“0”（偽値）とされた状態では、 $ADSP_i$ が“1”となっても、入力ライン $i(00) \sim i(07)$ に入力されているデータはラッチされない。

#### 【0081】

図7は図6の出力コントローラ35の内部構成を表すブロック図である。

#### 【0082】

出力コントローラ35は、4つのフリップ・フロップ（以下、「FF」という。）41～44からなるジョンソンカウンタ50、4つのAND回路45～48、及び1つのOR回路49によって構成されている。

#### 【0083】

FF41～44のクロック端子Cには、共通のクロック信号clockが入力されている。また、FF41, 42, 43, 44の出力 $Q_0, Q_1, Q_2, \text{NOT}(Q_3)$ は、それぞれ、FF42, 43, 44, 41のデータ入力 $D_1, D_2, D_3, D_0$ に接続されている。また、OR回路49は、リセット信号reset及び終了信号ENDの論理和のリセット信号reset' を出力する。FF41～44のリセット端子rstには、リセット信号reset' が入力されている。

#### 【0084】

このように構成することにより、ジョンソンカウンタ50の出力（ $Q_0, Q_1, Q_2, Q_3$ ）の状態は、クロック信号clockが立ち上がるごとに、

(0, 0, 0, 0)

→ (1, 0, 0, 0)

→ (1, 1, 0, 0)

→ (1, 1, 1, 0)

→ (1, 1, 1, 1)

→ (0, 1, 1, 1)

→ (0, 0, 1, 1)

→ (0, 0, 0, 1)

→ (0, 0, 0, 0)

のようにサイクリックに遷移する。すなわち、各クロックごとに、1ビット右に

シフトして、最上位のビット ( $Q_3$ ) を反転して、最下位のビット ( $Q_0$ ) とするという動作を繰り返す。

#### 【0085】

FF41, 43 の出力  $Q_0$ ,  $Q_2$  が、それぞれ、チップ・イネーブル信号  $CE_0$ ,  $CE_1$  として出力される。また、FF41, 43 の出力  $NOT(Q_0)$ ,  $NOT(Q_2)$  が、それぞれ、チップ・イネーブル信号  $CE_2$ ,  $CE_3$  として出力される。ここで、 $NOT(Q_i)$  は、出力  $Q_i$  の反転出力である。

#### 【0086】

また、AND回路45は、FF41～44の出力  $Q_0$ ,  $NOT(Q_1)$ ,  $NOT(Q_2)$ ,  $NOT(Q_3)$  の論理積をアドレス・ストロブ信号  $ADSP_0$  として出力する。AND回路46は、FF41～44の出力  $Q_0$ ,  $Q_1$ ,  $Q_2$ ,  $NOT(Q_3)$  の論理積をアドレス・ストロブ信号  $ADSP_1$  として出力する。AND回路47は、FF41～44の出力  $NOT(Q_0)$ ,  $Q_1$ ,  $Q_2$ ,  $Q_3$  の論理積をアドレス・ストロブ信号  $ADSP_2$  として出力する。AND回路48は、FF41～44の出力  $NOT(Q_0)$ ,  $NOT(Q_1)$ ,  $NOT(Q_2)$ ,  $Q_3$  の論理積をアドレス・ストロブ信号  $ADSP_3$  として出力する。

#### 【0087】

尚、論理関数メモリ4-0～4-3は、チップ・イネーブル信号  $CE_0$ ～ $CE_3$  が“0”とされた状態においては、低消費電力モードになるものとする。これにより、演算に使用されない論理関数メモリ4の消費電力が低減され、ルックアップテーブル・カスケード回路全体を、より低消費電力化することができる。

#### 【0088】

以上のように構成された本実施形態に係るルックアップテーブル・カスケード論理回路において、以下その動作について説明する。

#### 【0089】

図8は実施形態1に係るルックアップテーブル・カスケード論理回路の動作を表すフローチャート、図9は実施形態1に係るルックアップテーブル・カスケード論理回路の動作時における各信号の変化を表すタイミング図である。

#### 【0090】

最初に、演算を行おうとする目的論理関数  $f(X)$  を関数分解した分解関数  $f_0, \dots$

$\{f_{p-1}\}$  ( $1 \leq p$ ) の真理値表を、ルックアップテーブルとして各論理関数メモリ 4-0 ~ 4-3 に書き込んでおく。ここで、分解関数  $f_{4k} \sim f_{4k+3}$  ( $k=0, \dots, [(p-1)/4]$ ) のルックアップテーブルは、論理関数メモリ 4-0 ~ 4-3 の第  $k$  ページに書き込まれる。尚、同一の論理関数メモリ 4 において、異なるページに同一のルックアップテーブルが書き込まれることとなる場合には、複数ページに同一のルックアップテーブルを書き込むことはせず、1つのルックアップテーブルのみを書き込むようにする（以下、これを「書込縮約」という）。メモリ容量を節約するためである。また、入力選択メモリ 3-0 ~ 3-3 及び各接続メモリ 6-1 ~ 6-3 には、第 0 ページから順に上記各分解関数  $\{f_0, \dots, f_{p-1}\}$  に対応した入力選択情報及び接続情報を書き込んでおく。また、ページ選択メモリ 8- $i$  ( $i \in \{0, 1, 2, 3\}$ ) には、分解関数  $f_{4k+i}$  ( $k=0, \dots, [(p-1)/4]$ ) に対応するルックアップテーブルが記憶されている論理関数メモリ 4- $i$  のページ数  $p(4k+i)$  を、第  $[(k-1)/4]$  ページに格納する。尚、各メモリへの書き込み機能については、図 1 では図示していないが、通常のメモリの書き込み方法により行われる。更に、演算制御部 10 の演算ステップレジスタ 31 には、分解関数の全段数  $p$  を書き込んでおく。尚、以下では、 $p > 4$  であるものとして説明する。

#### 【0091】

以上の書き込みがされた状態において、まず、時間区間  $T_0$  において、演算制御部 10 は、外部から入力されるリセット信号  $reset$  によって、ステップカウンタ 32、ページカウンタ 34 のカウント値  $i$ 、 $k$  がそれぞれリセットされる (S1)。このとき、ステップカウンタ 32 のカウント値  $i$  は、演算ステップレジスタ 31 内に記憶された値  $p$  に初期化され、ページカウンタ 34 のカウント値  $k$  は 0 に初期化される。また、当該リセット信号  $reset$  により、演算制御部 10 内の出力コントローラ 35 の FF41 ~ 44 がリセットされる (S2)。

#### 【0092】

このとき、FF41 ~ 44 からなるジョンソンカウンタ 50 は、出力値として、 $(Q_0, Q_1, Q_2, Q_3) = (0, 0, 0, 0)$ 、 $(NOT(Q_0), NOT(Q_1), NOT(Q_2), NOT(Q_3)) = (1, 1, 1, 1)$  を出力する。従って、出力コントローラ 35 は、各論理関数メモリ 4-0, 4-1, 4-2, 4-3 に対して、チップ・イネーブル信号  $CE_0, CE_1, CE_2, CE$

3として、それぞれ0, 0, 1, 1を出力する。また、ページカウンタ34は、カウント信号pkとして0を、入力選択メモリ3-0～3-3、接続メモリ6-0～6-3、及びページ選択メモリ8-0～8-3に対して出力する（S3）。

#### 【0093】

これにより、入力選択メモリ3-i ( $i \in \{0, 1, 2, 3\}$ ) は、第0ページに格納された入力選択情報を、入力変数選択回路2-iの制御ラインshf0～shf3に出力する。入力変数選択回路2-iは、制御ラインshf0～shf3に輸入された入力選択情報に従って、入力端子in(00)～in(16)の一部を出力端子out(00)～out(07)と電気的に接続する。

#### 【0094】

また、接続メモリ6-i ( $i \in \{0, 1, 2, 3\}$ ) は、第0ページに格納された接続情報( $s_{Nc-1}^{(0)}, \dots, s_0^{(0)}$ )を接続回路5-iに出力する。これにより、接続回路5-iは、各制御ラインより入力される接続情報( $s_3^{(0)}, \dots, s_0^{(0)}$ )に従って、入力ラインi<sub>8</sub>～i<sub>15</sub>と出力ラインo<sub>0</sub>～o<sub>7</sub>とを接続する。

#### 【0095】

更に、ページ選択メモリ8-i ( $i \in \{0, 1, 2, 3\}$ ) は、第0ページに格納されたページ情報p(0)を論理関数メモリ4-iのページデコーダ13に出力する。これにより、論理関数メモリ4-iは、第p(0)ページが選択され得る状態となる。

#### 【0096】

そして、入力変数レジスタ1は、各入力変数選択回路2-0～2-3に対して、入力変数の出力を開始する（S4）。これにより、接続回路5-0～5-3には、それぞれ、入力変数X<sub>0</sub>, X<sub>1</sub>, X<sub>2</sub>, X<sub>3</sub>が出力される。接続回路5-i ( $i = 0, 1, 2, 3$ ) に入力された入力変数X<sub>0</sub>, X<sub>1</sub>, X<sub>2</sub>, X<sub>3</sub>は、設定された接続順序に従って、論理関数メモリ4-iの入力側に出力される。

#### 【0097】

続いて、時間区間T<sub>1</sub>において、次のクロック信号clockの立ち上がりで、ジョンソン・カウンタ50の出力(Q<sub>0</sub>, Q<sub>1</sub>, Q<sub>2</sub>, Q<sub>3</sub>)は、(1, 0, 0, 0)に変化する。これにより、CE<sub>0</sub>が“1”となり、論理関数メモリ4-0がアクセス可能な状態となるとともに、CE<sub>2</sub>は“0”となる。従って、論理関数メモリ4-0から、データD<sub>0</sub>が

出力され始める (S5)。

【0098】

また、これとほぼ同時に、AND回路45の出力であるADSP<sub>0</sub>が“1”となる。ADSP<sub>0</sub>が“1”となると、論理関数メモリ4-0は、接続回路5-0から入力される入力変数X<sub>0</sub>に基づいて、その内部のアドレスレジスタが制定される (S6)。そして、論理関数メモリ4-0は、第0ページ内の制定されたアドレスに対応するルックアップテーブルの値 (中間変数Y<sub>4k+(p-i)</sub>) を、接続回路5-1に出力し始める。接続回路5-1に出力された中間変数Y<sub>4k+(p-i)</sub>は、設定された接続関係に従って、論理関数メモリ4-1に入力される。また、場合によっては、中間変数Y<sub>4k+(p-i)</sub>の一部は、外部出力ライン7-1へと伝達される。

【0099】

時間区間T<sub>2</sub>において、ADSP<sub>0</sub>が“1”となってから次のクロック信号clockが立ち上がったときに、ジョンソン・カウンタ50の出力 (Q<sub>0</sub>, Q<sub>1</sub>, Q<sub>2</sub>, Q<sub>3</sub>) は、(1, 1, 0, 0) に変化する。これにより、ADSP<sub>0</sub>は“0”となる (S7)。これにより、論理関数メモリ4-0の出力データD<sub>0</sub>の値が中間変数Y<sub>4k+(p-i)</sub>に確定する。一方、このとき、Q<sub>0</sub>=1なので、CE<sub>0</sub>は“1”にラッチされている。

【0100】

次いで、時間区間T<sub>3</sub>において、次のクロック信号clockの立ち上がりで、ステップカウンタ32は、カウント値iをデクリメントする (S8)。また、このとき、ジョンソン・カウンタ50の値は(Q<sub>0</sub>, Q<sub>1</sub>, Q<sub>2</sub>, Q<sub>3</sub>)=(1, 1, 1, 0)となる。

【0101】

ここで、ステップカウンタ32のカウント値iが0でない場合において (S9)、出力Q<sub>2</sub>の値が1に遷移すると、CE<sub>1</sub>が“1”となるとともに、CE<sub>3</sub>が“0”となる。それに伴って、論理関数メモリ4-1の出力ラインから出力データD<sub>1</sub>が出力され始める (S10)。

【0102】

また、これとほぼ同時に、AND回路46の出力であるADSP<sub>1</sub>が“1”となる (S11)。これにより、論理関数メモリ4-1は、接続回路5-1を介して前段の論理関数メモリ4-0から入力される中間変数と新たな入力変数とに基づいてアド

レスが制定される。このアドレスの制定に伴って、論理関数メモリ 4-1 の出力データ  $D_1$  が変化し、第  $pk$  ページ内の制定されたアドレスに対応するルックアップテーブルの値（中間変数  $Y_{4k+(p-i)}$ ）が、接続回路 5-2 に出力され始める。

#### 【0103】

そして、時間区間  $T_4$  において、次のクロック信号  $clock$  の立ち上がりで、ジョンソン・カウンタ 50 の値は  $(Q_0, Q_1, Q_2, Q_3) = (1, 1, 1, 1)$  となる。これにより、AND 回路 46 の出力である  $ADSP_1$  が “0” となり、論理関数メモリ 4-1 の出力データ  $D_1$  が、中間変数  $Y_{4k+(p-i)}$  に確定する（S12）。接続回路 5-2 に出力された中間変数  $Y_{4k+(p-i)}$  は、設定された接続関係に従って、論理関数メモリ 4-2 に入力される。また、場合によっては、中間変数  $Y_{4k+(p-i)}$  の一部は、外部出力ライン 7-2 へと伝達される。

#### 【0104】

次いで、時間区間  $T_5$  において、次のクロック信号  $clock$  の立ち上がりで、ステップカウンタ 32 は、カウント値  $i$  をデクリメントする（S13）。また、このとき、ジョンソン・カウンタ 50 の値は  $(Q_0, Q_1, Q_2, Q_3) = (0, 1, 1, 1)$  となる。

#### 【0105】

ここで、ステップカウンタ 32 のカウント値  $i$  が 0 でない場合（S14）、出力  $Q_0$  の値が 0 に遷移すると、 $CE_2$  が “1” となるとともに、 $CE_0$  は “0” となる。それに伴って、論理関数メモリ 4-0 の出力が停止するとともに、論理関数メモリ 4-2 の出力ラインから出力データ  $D_2$  が出力され始める（S15）。

#### 【0106】

また、これとほぼ同時に、AND 回路 47 の出力である  $ADSP_2$  が “1” となる（S16）。これにより、論理関数メモリ 4-2 は、接続回路 5-2 を介して前段の論理関数メモリ 4-1 から入力される中間変数と新たな入力変数とに基づいてアドレスが制定される。このアドレスの制定に伴って、論理関数メモリ 4-2 の出力データ  $D_2$  が変化し、第  $pk$  ページ内の制定されたアドレスに対応するルックアップテーブルの値（中間変数  $Y_{4k+(p-i)}$ ）が、接続回路 5-3 に出力され始める。

#### 【0107】

そして、時間区間  $T_6$  において、次のクロック信号  $clock$  の立ち上がりで、ジョ

ンソン・カウンタ 50 の値は  $(Q_0, Q_1, Q_2, Q_3) = (0, 0, 1, 1)$  となる。これにより、AND 回路 47 の出力である  $ADSP_2$  が “0” となり、論理関数メモリ 4-2 の出力データ  $D_2$  が、中間変数  $Y_{4k+(p-i)}$  に確定する (S 17)。接続回路 5-3 に出力された中間変数  $Y_{4k+(p-i)}$  は、設定された接続関係に従って、論理関数メモリ 4-3 に入力される。また、場合によっては、中間変数  $Y_{4k+(p-i)}$  の一部は、外部出力ライン 7-3 へと伝達される。

#### 【0108】

次いで、時間区間  $T_7$  において、次のクロック信号  $clock$  の立ち上がりで、ステップカウンタ 32 は、カウント値  $i$  をデクリメントする (S 18)。また、このとき、ジョンソン・カウンタ 50 の値は  $(Q_0, Q_1, Q_2, Q_3) = (0, 0, 0, 1)$  となる。

#### 【0109】

ここで、ステップカウンタ 32 のカウント値  $i$  が 0 でない場合において (S 19)、出力  $Q_2$  の値が 0 に遷移すると、 $CE_3$  が “1” となるとともに、 $CE_1$  は “0” となる。それに伴って、論理関数メモリ 4-1 の出力が停止するとともに、論理関数メモリ 4-3 の出力ラインから出力データ  $D_3$  が出力され始める (S 20)。

#### 【0110】

また、これとほぼ同時に、AND 回路 48 の出力である  $ADSP_3$  が “1” となる (S 21)。これにより、論理関数メモリ 4-3 は、接続回路 5-3 を介して前段の論理関数メモリ 4-2 から入力される中間変数と新たな入力変数とに基づいてアドレスが制定される。このアドレスの制定に伴って、論理関数メモリ 4-3 の出力データ  $D_3$  が変化し、第  $p_k$  ページ内の制定されたアドレスに対応するルックアップテーブルの値 (中間変数  $Y_{4k+(p-i)}$ ) が、接続回路 5-0 に出力され始める。

#### 【0111】

そして、時間区間  $T_0$  において、次のクロック信号  $clock$  の立ち上がりで、ジョンソン・カウンタ 50 の値は  $(Q_0, Q_1, Q_2, Q_3) = (0, 0, 0, 0)$  となる。これにより、AND 回路 48 の出力である  $ADSP_3$  が “0” となり、論理関数メモリ 4-3 の出力データ  $D_3$  が、中間変数  $Y_{4k+(p-i)}$  に確定する (S 22)。接続回路 5-0 に出力された中間変数  $Y_{4k+(p-i)}$  は、設定された接続関係に従って、論理関数メモリ 4-0 に入力される。また、場合によっては、中間変数  $Y_{4k+(p-i)}$  の一部は、外部出力ライン

7-0へと伝達される。

#### 【0112】

このとき、ページカウンタ34は、カウント値 $k$ をインクリメントする(S23)。これにより、カウント信号 $pk$ の値が1だけ増加し、入力選択メモリ3-0～3-3、接続メモリ6-0～6-3、及びページ選択メモリ8-0～8-3の選択されるページが切り替わる。これに伴って、入力変数選択回路2-0～2-3及び接続回路5-0～5-3の接続関係も切り替わる。尚、ページ選択メモリ8-0～8-3の選択されるページが切り替わった場合であっても、各論理関数メモリは、チップ・イネーブル信号 $CE_i$ 及びアドレス・ストロブ信号 $ADSP_i$ が“1”となってアドレスを受け付ける状態とならない限りにおいては、ページが切り替わることはない、

#### 【0113】

次いで、時間間隔 $T_1$ において、次のクロック信号 $clock$ の立ち上がりで、ステップカウンタ32は、カウント値 $i$ をデクリメントする(S24)。また、このとき、ジョンソン・カウンタ50の出力 $Q_0$ の値が1となる。

ここで、ステップカウンタ32のカウント値 $i$ が0でない場合には(S25)、再びステップS5の動作に戻る。

#### 【0114】

上記ステップS9、S14、S19、S25において、ステップカウンタ32のカウント値 $i$ が0であった場合は、ステップカウンタ32は終了信号ENDを出力する。これにより、演算制御部10は演算動作を停止し、演算が終了する。

#### 【0115】

以上のように、演算処理は、前段から後段に向かって直列処理により行われる。そして、最後段の論理関数メモリ4-3における演算処理が終了した時点で、まだ総ての分解関数の演算が終了していない場合には、論理関数メモリ4-3から出力される中間変数を再び最前段の論理関数メモリにフィードバックして、演算処理を繰り返す。そして、目的関数 $f$ の演算結果は、各外部出力ライン7-1～7-4より取り出される。

#### 【0116】



尚、本実施形態では接続回路 5-1 ~ 5-3 として、図 4 のシフタ回路を使用したが、接続回路 5-1 ~ 5-3 としては、図 10 に示したクロスバスイッチ 5-i' や図 11 に示したマルチプレクサ 5-i'' を使用することも可能である。特に、クロスバスイッチ 5-i' は、シフタ回路に比べると切換制御ラインの数が多くなり回路が大きくなるという欠点はあるが、信号が通過するバストランジスタの段数を減らすことができるため、演算速度の高速化が可能となる。

#### 【0117】

次に、上記動作をより分かりやすく説明するため、具体的な例を用いてルックアップテーブル・カスケード論理回路の具体的な動作を説明する。

#### 【0118】

##### (例) 加算回路

ここでは、簡単な例として、二つの  $2n$  ビット ( $n \geq 5$ ) の二進数  $A = (a_{2n-1}, a_{2n-2}, \dots, a_1, a_0)$ ,  $B = (b_{2n-1}, b_{2n-2}, \dots, b_1, b_0)$  の加算を行う加算器を、上記ルックアップテーブル・カスケード論理回路によって実行する例を説明する。尚、下位からの桁上げ (キャリー) 入力ビットとして  $c_{in}$  も考慮して、以下のような演算を行う加算器を考える。

#### 【数 1】

$$\begin{array}{r}
 a_{2n-1} \quad a_{2n-2} \quad \cdots \quad a_1 \quad a_0 \\
 b_{2n-1} \quad b_{2n-2} \quad \cdots \quad b_1 \quad b_0 \\
 +) \\
 \hline
 c_{out} \quad S_{2n-1} \quad S_{2n-2} \quad \cdots \quad S_1 \quad S_0
 \end{array}$$

ここで、桁上げ入力ビット  $c_{in}$  とは、 $2n$  ビット以上の大きな数の加算を行う場合に、加算器を直列につなげて使用する際に、下位の加算器からの桁上げを表すビットである。桁上げ出力ビット  $c_{out}$  も、同様に、上位の加算器に出力する桁上げを表すビットである。

#### 【0119】

尚、この例においては、入力変数選択回路 2-0 ~ 2-3 は、図 2 と同様に、 $\log_2 x$  ( $\log_2 x$  が整数でないときは、 $\log_2 x$  を切り上げた数) 段のシフタが直列接続された、 $x$  入力 8 出力のシフタ回路で構成されているものとし、 $2n+1 \leq x$  である

ものとする。

【0120】

このような二つの  $2n$  ビットの二進数  $A, B$  の加算を行う論理関数を  $f=f(X)$  ( $X=(A, B)$ ) とすると、論理関数  $f$  は図 12 により表される。この関数は、図 13 のように、 $2n$  個の分解関数  $\{g_0, g_1, \dots, g_{2n-1}\}$  に関数分解することが可能である。ここで、各分解関数は、(数 2) 又は (数 3) のような論理式で表される。

【数 2】

$$\begin{aligned} g_0 &= [c_{out}^{(0)}, S_0] \\ &= [a_0 b_0 \vee a_0 c_{in} \vee b_0 c_{in}, a_0 \oplus b_0 \oplus c_{in}] \end{aligned}$$

【数 3】

$$\begin{aligned} g_i &= [c_{out}^{(i)}, S_i] \\ &= [a_i b_i \vee a_i c_{out}^{(i-1)} \vee b_i c_{out}^{(i-1)}, a_i \oplus b_i \oplus c_{out}^{(i-1)}] \\ &\quad (i = 1, 2, \dots, 2n-1) \end{aligned}$$

すなわち、各分解関数は、2つの入力変数  $a_i, b_i$  及び桁上げを表す中間変数  $c_{out}^{(i-1)}$  の 2 を法とする和  $S_i$  及び桁上げの中間変数  $c_{out}^{(i)}$  を出力する論理関数となっている。

【0121】

これを、図 1 のような 4 つの論理関数メモリ 4-0 ~ 4-3 からなるルックアップテーブル・カスケード論理回路で実現する場合には、図 14 に示したように、分解関数  $\{g_0, g_1, \dots, g_{2n-1}\}$  を、 $n$  個の分解関数組  $\{g_0, g_1\}, \{g_2, g_3\}, \dots, \{g_{2n-2}, g_{2n-1}\}$  に分ける。そして、図 15 に示したように、各々の分解関数組を 1 つにまとめて  $n$  個の分解関数  $f_0, f_1, \dots, f_{n-1}$  で表す。ここで、各分解関数は、(数 4) 又は (数 5) のような論理式で表される 3 変数入力 3 変数出力論理関数である。

## 【数 4】

$$\begin{aligned}
 f_0 &= [c_{out}^{(0)}, S_1, S_0] \\
 &= [a_1 b_1 \vee (a_1 \vee b_1)(a_0 b_0 \vee a_0 c_{in} \vee b_0 c_{in}), \\
 &\quad a_1 \oplus b_1 \oplus (a_0 b_0 \vee a_0 c_{in} \vee b_0 c_{in}), a_0 \oplus b_0 \oplus c_{in}]
 \end{aligned}$$

## 【数 5】

$$\begin{aligned}
 f_i &= [c_{out}^{(i)}, S_{2i+1}, S_{2i}] \\
 &= [a_{2i+1} b_{2i+1} \vee (a_{2i+1} \vee b_{2i+1})(a_{2i} b_{2i} \vee a_{2i} c_{out}^{(i-1)} \vee b_{2i} c_{out}^{(i-1)}), \\
 &\quad a_{2i+1} \oplus b_{2i+1} \oplus (a_{2i} b_{2i} \vee a_{2i} c_{out}^{(i-1)} \vee b_{2i} c_{out}^{(i-1)}), a_{2i} \oplus b_{2i} \oplus c_{out}^{(i-1)}] \\
 &\quad (i = 1, 2, \dots, n-1)
 \end{aligned}$$

## 【0 1 2 2】

各々の分解関数 $f_i$ の真理値表は、図16 (a) のようになる。そこで、図16 (b) の真理値表を、論理関数メモリ4-0の第1ページに記憶させ、図16 (c) の真理値表(ルックアップテーブル)を論理関数メモリ4-0～4-3の第0ページに記憶させる。ここで、それぞれ、 $S_{2i}$ ,  $S_{2i+1}$ ,  $c_{out}^{(i)}$  ( $i=1, 2, \dots, n-1$ ) は、それぞれ接続回路5-( $i+1$ )の入力ライン $i_2, i_3, i_4$ に出力されるように論理関数メモリの出力ビットを割り当てる。

## 【0 1 2 3】

各入力選択メモリ3-0～3-3の第0ページには、それぞれ、0ビットシフト、5ビットシフト、9ビットシフト、13ビットシフト、第 $k$ ページ( $1 \leq k \leq [(n-1)/4]$ )には、それぞれ、 $16k+1$ ビットシフト、 $16k+5$ ビットシフト、 $16k+9$ ビットシフト、 $16k+13$ ビットシフトの情報を記憶させておく。

## 【0 1 2 4】

接続メモリ6-0の第0ページには、接続情報として、8ビット分シフトさせる情報を記憶させる。接続メモリ6-1～6-3の第0ページには、接続情報として4ビット分シフトさせる情報を記憶させる。また、各接続メモリ6-0～6-3の第 $k$ ページ( $1 \leq k \leq [(n-1)/4]$ )には、接続情報として4ビット分シフトさせる情報を記憶させる。但し、接続メモリ6- $\alpha$  ( $\alpha = n \bmod 4$ )の第 $[n/4]$ ページの接続情報は、最終段の分解関数の出力変数(5ビット)を出力させるた

めの接続情報なので、当該ページには5ビット分シフトさせる情報を記憶させる。尚、4ビット分シフトさせる場合には接続情報として $(s_3, s_2, s_1, s_0) = (0, 1, 0, 0)$ を、5ビット分シフトさせる場合には $(s_3, s_2, s_1, s_0) = (0, 1, 0, 1)$ を、8ビット分シフトさせる場合には接続情報として $(s_3, s_2, s_1, s_0) = (1, 0, 0, 0)$ を接続メモリの各ページに記憶させる。例えば、 $n = 13$ の場合、接続メモリ6の内容は、以下の(表1)のようになる。

【表1】

ページ	接続メモリ 6-0	接続メモリ 6-1	接続メモリ 6-2	接続メモリ 6-3
0	(1, 0, 0, 0)	(0, 1, 0, 0)	(0, 1, 0, 0)	(0, 1, 0, 0)
1	(0, 1, 0, 0)	(0, 1, 0, 0)	(0, 1, 0, 0)	(0, 1, 0, 0)
2	(0, 1, 0, 0)	(0, 1, 0, 0)	(0, 1, 0, 0)	(0, 1, 0, 0)
3	(0, 1, 0, 0)	(0, 1, 0, 1)		

【0125】

ページ選択メモリ8-0の第0ページには、ページ情報として1を記憶させておく。また、ページ選択メモリ8-1～8-3の第0ページには、ページ情報として0を記憶させておく。更に、ページ選択メモリ8-0～8-3の第kページ( $1 \leq k \leq [(n-1)/4]$ )には、ページ情報として0を記憶させておく。例えば、 $n = 13$ の場合、接続メモリ6の内容は、以下の(表2)のようになる。

【表2】

ページ	接続メモリ 6-0	接続メモリ 6-1	接続メモリ 6-2	接続メモリ 6-3
0	1	0	0	0
1	0	0	0	0
2	0	0	0	0
3	0			

【0126】

以上のようにプログラムされた状態において、以下、ルックアップテーブル・カスケード論理回路の演算処理を説明する。尚、以下の説明では、簡単のため  $n = 5$  として説明する。

#### 【0127】

まず、入力変数レジスタ 1 には、変数  $A, B, c_{in}$  が入力され記憶される。また、演算ステップレジスタ 31 には、 $n = 5$  が記憶される。そして、演算制御部 10 は、まず、ステップカウンタ 32 のカウント値  $i$  を  $n = 5$  にリセットし、ページカウンタ 34 のカウント値  $k$  を 0 にリセットする。更に、ジョンソン・カウンタ 50 も 0 にリセットする。これにより、演算制御部 10 のページカウンタ 34 は、入力選択メモリ 3-0 ~ 3-3、接続メモリ 6-0 ~ 6-3、及びページ選択メモリ 8-0 ~ 8-3 に対してページ選択番号  $pk$  として第 0 ページを設定する。

#### 【0128】

また、入力変数選択回路 2-0 ~ 2-3 は、各々の制御ラインに入力された入力選択情報に従って、入力端子  $in(00) \sim in(x-1)$  のうちの 8 つを出力端子  $out(00) \sim out(07)$  と電氣的に接続する。

#### 【0129】

具体的には、入力選択メモリ 3-0 は、 $in(00) \sim in(07)$  を  $out(00) \sim out(07)$  に接続し、入力選択メモリ 3-1 は、5 ビットシフトさせて、 $in(05) \sim in(12)$  を  $out(00) \sim out(07)$  に接続し、入力選択メモリ 3-2 は、9 ビットシフトさせて、 $in(09) \sim in(16)$  を  $out(00) \sim out(07)$  に接続し、入力選択メモリ 3-3 は、13 ビットシフトさせて、 $in(13) \sim in(20)$  を  $out(00) \sim out(07)$  に接続する。

#### 【0130】

また、接続メモリ 6-0 ~ 6-3 は、各々、第 0 ページに書き込まれた接続情報 ( $s_3^{(0)}, \dots, s_0^{(0)}$ ) を接続回路 5-0 ~ 5-3 の各制御ラインに出力する。接続回路 5-0 ~ 5-3 は、各制御ラインより入力される接続情報 ( $s_3^{(0)}, \dots, s_0^{(0)}$ ) に従って、入力ライン  $i_8 \sim i_{15}$  と出力ライン  $o_0 \sim o_7$  とを接続する。具体的には、接続回路 5-0 は、8 ビットシフトさせて、 $i_8 \sim i_{15}$  をそれぞれ  $o_0 \sim o_7$  に接続し、 $i_0 \sim i_7$  をそれぞれ  $o_8 \sim o_{15}$  に接続する。また、接続メモリ 6-1 ~ 6-3 は、4 ビットシフトさせて、 $i_4 \sim i_{11}$  をそれぞれ  $o_0 \sim o_7$  に接続し、 $i_{12} \sim i_{15}$ ,  $i_0 \sim i_3$  をそれぞれ  $o_8 \sim o_{15}$  に接続する。

8～0<sub>15</sub>に接続する。

### 【0131】

また、ページ選択メモリ 8-0～8-3は、各々、第0ページに書き込まれたページ情報を、論理関数メモリ 4-0～4-3のページデコーダ15に対して出力する。これにより、論理関数メモリ 4-0は、1ページがアクセス可能な状態となり、論理関数メモリ 4-1～4-3は、1ページがアクセス可能な状態となる。

### 【0132】

次に、演算制御部10は、入力変数レジスタ1により、入力変数Xの出力を行う。このとき、入力変数選択回路2-1～2-3の入力端子には、それぞれ、(表3)のように入力変数の値が入力される。

【表3】

i(00)	i(01)	i(02)	i(03)	i(04)	i(05)	i(06)	i(07)	i(08)
c <sub>in</sub>	a <sub>1</sub>	a <sub>0</sub>	b <sub>1</sub>	b <sub>0</sub>	a <sub>3</sub>	a <sub>2</sub>	b <sub>3</sub>	b <sub>2</sub>
i(09)	i(10)	i(11)	i(12)	i(13)	i(14)	i(15)	i(16)	i(17)
a <sub>5</sub>	a <sub>4</sub>	b <sub>5</sub>	b <sub>4</sub>	a <sub>7</sub>	a <sub>6</sub>	b <sub>7</sub>	b <sub>6</sub>	a <sub>9</sub>
i(18)	i(19)	i(20)						
a <sub>10</sub>	b <sub>9</sub>	b <sub>10</sub>						

### 【0133】

入力変数選択回路2-0は、(out(00), out(01), out(02), out(03), out(04)) = (c<sub>in</sub>, a<sub>1</sub>, a<sub>0</sub>, b<sub>1</sub>, b<sub>0</sub>)を出力し、これを接続回路5-0の入力ライン(i<sub>8</sub>, i<sub>9</sub>, i<sub>10</sub>, i<sub>11</sub>, i<sub>12</sub>)に入力する。尚、出力out(05)～out(07)は使用されないので省略した。接続回路5-0は、8ビットシフトさせて、これらを出力ライン(o<sub>0</sub>, o<sub>1</sub>, o<sub>2</sub>, o<sub>3</sub>, o<sub>4</sub>)に出力する。このようにして、入力変数(c<sub>in</sub>, a<sub>1</sub>, a<sub>0</sub>, b<sub>1</sub>, b<sub>0</sub>)が論理関数メモリ4-0に入力される。

### 【0134】

次に、演算制御部10の出力コントローラ35は、CE<sub>0</sub>を“1”に設定して論理関数メモリ4-0のデータ出力を行わせるとともに、ADSP<sub>0</sub>を“1”に設定して、論理関数メモリ4-0内部のアドレスレジスタに、アドレス(i<sub>0</sub>, i<sub>1</sub>, i<sub>2</sub>, i<sub>3</sub>,

$i_4, i_5, i_6, i_7) = (c_{in}, a_1, a_0, b_1, b_0, -, -, -)$  を制定する。ここで、「-」はドントケア（0でも1でもよい。）を表す。アドレス制定後に、出力コントローラ35は、ADSP<sub>0</sub>を“0”とする。

#### 【0135】

アドレス制定後、論理関数メモリ4-0の出力端子には、第1ページに格納されたルックアップテーブルLUT<sub>1</sub>のアドレス  $(c_{in}, a_1, a_0, b_1, b_0, -, -, -)$  に記憶された真理値  $f_0(c_{in}, a_1, a_0, b_1, b_0)$  の値  $(-, -, S_0, S_1, c_{out}^{(0)}, -, -, -)$  が現れ、これが接続回路5-1に出力される。すなわち、接続回路5-1の各入力ライン  $(i_0, i_1, i_2, i_3, i_4, i_5, i_6, i_7)$  には  $(-, -, S_0, S_1, c_{out}^{(0)}, -, -, -)$  が入力される。

#### 【0136】

一方、入力変数選択回路2-1は、 $(out(00), out(01), out(02), out(03)) = (a_3, a_2, b_3, b_2)$  を出力し、これを接続回路5-1の入力ライン  $(i_8, i_9, i_{10}, i_{11})$  に入力する。

#### 【0137】

接続回路5-1からは、それぞれ、(表4)の値が出力される。

【表4】

$o_0$	$o_1$	$o_2$	$o_3$	$o_4$	$o_5$	$o_6$	$o_7$
$c_{out}^{(0)}$	-	-	-	$a_3$	$a_2$	$b_3$	$b_2$
$o_8$	$o_9$	$o_{10}$	$o_{11}$	$o_{12}$	$o_{13}$	$o_{14}$	$o_{15}$
-	-	-	-	-	-	$S_0$	$S_1$

#### 【0138】

接続回路5-1の出力ライン $o_8 \sim o_{15}$ は、外部出力ライン7-1として取り出される。従って、演算結果 $S_0, S_1$ が外部出力ライン7-1のうちの2本に出力される。

#### 【0139】

次に、演算制御部10のステップカウンタ32は、そのカウント値 $i$ を $i-1 = 4$ に更新する。そして、演算制御部10の出力コントローラ35は、CE<sub>1</sub>を“1”に設定して論理関数メモリ4-1のデータ出力を行わせるとともに、ADSP<sub>1</sub>を

“1” に設定して、論理関数メモリ 4-1 内部のアドレスレジスタに、アドレス( $i_0, i_1, i_2, i_3, i_4, i_5, i_6, i_7$ ) = ( $c_{out}^{(0)}, -, -, -, a_3, a_2, b_3, b_2$ ) を制定する。アドレス制定後に、出力コントローラ 35 は、ADSP<sub>1</sub> を “0” とする。更に、0 段目の論理関数メモリ 4-0 の出力値は既にアドレスレジスタに取り込んだので、出力コントローラ 35 は、CE<sub>0</sub> を “0” として、論理関数メモリ 4-0 の出力を停止させる。

#### 【0140】

アドレス制定後、論理関数メモリ 4-1 の出力端子には、第 0 ページに格納されたルックアップテーブル LUT<sub>0</sub> のアドレス ( $c_{out}^{(0)}, -, -, -, a_3, a_2, b_3, b_2$ ) に記憶された真理値  $f_1(c_{out}^{(0)}, a_3, a_2, b_3, b_2)$  の値 ( $-, -, S_2, S_3, c_{out}^{(1)}, -, -, -$ ) が現れ、これが接続回路 5-2 に出力される。すなわち、接続回路 5-2 の各入力ライン ( $i_0, i_1, i_2, i_3, i_4, i_5, i_6, i_7$ ) には ( $-, -, S_2, S_3, c_{out}^{(1)}, -, -, -$ ) が入力される。

#### 【0141】

一方、入力変数選択回路 2-2 は、( $out(00), out(01), out(02), out(03)$ ) = ( $a_5, a_4, b_5, b_4$ ) を出力し、これを接続回路 5-2 の入力ライン ( $i_8, i_9, i_{10}, i_{11}$ ) に入力する。

#### 【0142】

接続回路 5-2 からは、それぞれ、(表 5) の値が出力される。

【表 5】

$o_0$	$o_1$	$o_2$	$o_3$	$o_4$	$o_5$	$o_6$	$o_7$
$c_{out}^{(1)}$	-	-	-	$a_5$	$a_4$	$b_5$	$b_4$
$o_8$	$o_9$	$o_{10}$	$o_{11}$	$o_{12}$	$o_{13}$	$o_{14}$	$o_{15}$
-	-	-	-	-	-	$S_2$	$S_3$

#### 【0143】

接続回路 5-2 の出力ライン  $o_8 \sim o_{15}$  は、外部出力ライン 7-2 として取り出される。従って、演算結果  $S_2, S_3$  が外部出力ライン 7-2 のうちの 2 本に出力される。



## 【0144】

以下同様にして、論理関数メモリ 4-2、論理関数メモリ 4-3 においても演算が行われる。そして、論理関数メモリ 4-2 の演算結果  $S_4, S_5$  は外部出力ライン 7-3 のうちの 2 本に出力される。また、論理関数メモリ 4-3 の出力値  $(-, -, S_6, S_7, c_{out}^{(3)}, -, -, -)$  は、接続回路 5-0 の入力ライン  $i_0 \sim i_7$  にフィードバックされる。

## 【0145】

次に、演算制御部 10 のステップカウンタ 32 は、そのカウント値  $i$  を  $i-1 = 1$  に更新する。ここで、論理関数メモリ 4-3 における演算処理が終了したので、ページカウンタ 34 は、そのカウント値  $k$  を 1 だけインクリメントし、そのカウント値  $k$  をカウント信号  $pk$  に出力する。

## 【0146】

これに伴い、入力選択メモリ 3、接続メモリ 6、及びページ選択メモリ 8 のアクセスページが変更されるため、これらのメモリからの出力も変更される。具体的には、入力選択メモリ 3-0 から出力されるシフト量の情報が 17 ビットシフトに変更される。また、接続メモリ 5-0 から出力される接続情報が  $(s_3, s_2, s_1, s_0) = (0, 1, 0, 0)$  に変更され、接続メモリ 5-1 から出力される接続情報が  $(s_3, s_2, s_1, s_0) = (0, 1, 0, 1)$  に変更される。更に、ページ選択メモリ 8-0 が出力するページ情報  $p(pk=1)$  の値が 0 に変更され、論理関数メモリ 4-0 は 0 ページがアクセス可能な状態となる。

## 【0147】

また、これにより、論理関数メモリ 4-3 の演算結果  $S_6, S_7$  は外部出力ライン 7-0 のうちの 2 本に出力される。

## 【0148】

そして、上述の場合と同様に、論理関数メモリ 4-0 において演算処理が行われ、論理関数メモリ 4-0 の出力値  $(-, -, S_8, S_9, c_{out}^{(4)}, -, -, -)$  は接続回路 5-1 の入力ライン  $i_0 \sim i_7$  に入力される。接続回路 5-1 は、これを 5 ビットシフトして、 $(S_8, S_9, c_{out}^{(4)})$  を出力ライン  $(o_{13}, o_{14}, o_{15})$  に出力する。これにより、論理関数メモリ 4-0 の演算結果  $(S_8, S_9, c_{out}^{(4)})$  は外部出力ライン 7-1

のうちの3本に出力される。以上で、総ての演算結果が出力ライン7-0～7-3に出力されて演算が終了する。

#### 【0149】

尚、上記例においては、各段のルックアップテーブルに入力する入力変数 $X_0$ ,  $X_1$ , ...,  $X_{p-1}$ は互いに共通の要素を持たない（すなわち、 $X_i \cap X_j = \emptyset$  ( $i \neq j$ )）と仮定したが、本発明におけるルックアップテーブル・カスケード論理回路は、入力変数 $X_0$ ,  $X_1$ , ...,  $X_{p-1}$ の何れか2つが共通の要素を有するような場合にも使用することが可能である。

#### 【0150】

##### （実施形態2）

図17は本発明の実施形態2に係るルックアップテーブル・カスケード論理回路の全体構成を表すブロック図である。

#### 【0151】

本実施形態においては、接続回路5-0～5-3に入力する入力変数を選択する入力変数選択回路2-0～2-3に加えて、接続回路5-0～5-3を介さずに、論理関数メモリ4-0～4-3に直接入力する入力変数の選択を行う入力変数選択回路60-0～60-3及びその入力選択情報を記憶する入力選択メモリ61-1～61-3を設けたことを第1の特徴とする。また、前段の論理関数メモリ4-（ $i-1$ ）（ $i \in \{1, 2, 3\}$ ）から後段の論理関数メモリ4- $i$ に入力される中間変数の一部を、接続回路5- $i$ を介さずに直接入力するように構成したことを第2の特徴とする。

#### 【0152】

このように構成することで、接続回路5-0～5-3の入力ラインの本数を減らすことができる。その結果、接続回路5-0～5-3を小型化することが可能となる。また、接続回路5-0～5-3を図4のようなシフタ回路により構成する場合には、シフタの段数を減らすことが可能であり、演算速度の高速化を図ることができる。

#### 【0153】

##### （実施形態3）

図18は本発明の実施形態3に係るルックアップテーブル・カスケード論理回路の全体構成を表すブロック図である。

#### 【0154】

本実施形態においては、フィードバック用の接続回路の入力側に、最後段の論理関数メモリ4-3の複数個の出力を一時的に記憶し、フィードバック用の接続回路5-0に出力する中間変数バッファ65を備えていることを特徴とする。

#### 【0155】

実施形態1においては、論理関数メモリ4-0～4-3として、クロックに同期して動作するシンクロナスなメモリを使用した。しかし、論理関数メモリ4-0～4-3は、シンクロナスなメモリには限られず、非同期に動作するメモリを使用することも可能である。

#### 【0156】

しかしながら、最終段の論理関数メモリ4-3の出力は、最前段の論理関数メモリ4-0の入力にフィードバックされることから、論理関数メモリ4-0～4-3に非同期のメモリを使用した場合、場合によっては発振を起こすおそれがある。

#### 【0157】

そこで、本実施形態では、中間変数バッファ65を備えた構成とし、1演算ループの演算が終了するごとに、論理関数メモリ4-0～4-3の出力値を一旦確定させ、論理関数メモリ4-3が出力する中間変数の値を中間変数バッファ65に記憶させる。そして、次の演算ループに移り、中間変数バッファ65に記憶された中間変数を値の論理関数メモリ4-0に入力して演算処理を行うようにする。

#### 【0158】

これにより、論理関数メモリ4-0～4-3に非同期なメモリを使用した場合にも、発振することを防止できる。

#### 【0159】

(実施形態4)

図19は本発明の実施形態4に係るルックアップテーブル・カスケード論理回路の全体構成を表すブロック図、図20は図19の出力レジスタ及び出力デコー

ダの構成を表すブロック図、図 21 は図 20 の記憶素子の構成を表すブロック図である。本実施形態においては、外部出力ライン 7-1 ~ 7-3 に出力された出力変数を一時的に記憶する出力変数レジスタ 71、及び出力変数レジスタ 71 が出力変数を取り込むためのロード信号を出力する出力選択デコーダ 72 を備えたことを特徴とする。他の構成については、図 1 と同様であるため、説明は省略する。尚、本実施形態においても、各論理関数メモリ 4-0 ~ 4-3 はクロックにより同期して動作するものとする。

#### 【0160】

出力変数レジスタ 71 は、32 個の記憶素子  $M(i, j)$  ( $i=1, 2, 3, 4, j=0, 1, 2, 3, 4, 5, 6, 7$ ) を備えている。各記憶素子  $M(i, j)$  は、データを保持する D フリップ・フロップ 77 と 2 入力 1 出力のマルチプレクサ 78 とから構成されている。D フリップ・フロップ 77 には、共通のクロック信号 Clock が入力される。このクロック信号 Clock が 1 のときに、D フリップ・フロップ 77 はデータ入力 D に入力された値をラッチする。マルチプレクサ 78 の出力は D フリップ・フロップ 77 のデータ入力 D に接続されている。また、D フリップ・フロップ 77 の出力 Q は、マルチプレクサ 78 の 0 側の入力 D0 に接続されている。マルチプレクサ 78 の 1 側の入力 D1 は、外部出力ライン 7-i の j 番目のラインに接続されている。マルチプレクサ 78 は、ロード信号 Load により制御され、Load = 0 のときに 0 側の入力を選択し、Load = 1 のときに 1 側の入力を選択する。

#### 【0161】

出力選択デコーダ 72 には、2 ビットの出力選択信号  $t$  が演算制御部 10 から入力され、4 つの出力選択信号  $T_1 \sim T_4$  を出力する。 $t = (0, 0)$  のときは、 $(T_1, T_2, T_3, T_4) = (1, 0, 0, 0)$ 、 $t = (0, 1)$  のときは、 $(T_1, T_2, T_3, T_4) = (0, 1, 0, 0)$ 、 $t = (1, 0)$  のときは、 $(T_1, T_2, T_3, T_4) = (0, 0, 1, 0)$ 、 $t = (1, 1)$  のときは、 $(T_1, T_2, T_3, T_4) = (0, 0, 0, 1)$  が出力される。各出力選択信号  $T_i$  ( $i = 1, 2, 3, 4$ ) は、各記憶素子  $M(i, j)$  にロード信号 Load として入力される。

#### 【0162】

演算制御部 10 は、演算開始時には、 $t = (0, 0)$  として、論理関数メモリ 4

-0 が演算結果を出力した後に  $t = (0\ 1)$ 、論理関数メモリ 4-1 が演算結果を出力した後に  $t = (1\ 0)$ 、論理関数メモリ 4-2 が演算結果を出力した後に  $t = (1\ 1)$  のように出力選択信号  $t$  を切り換えて、演算結果を出力変数レジスタ 71 にラッチする。そして、演算が終了した時点で、出力変数レジスタ 71 に記憶された出力変数を読み出すことで、演算結果を得ることができる。

### 【0163】

(実施形態 5)

図 22 は本発明の実施形態 5 に係るルックアップテーブル・カスケード論理回路の全体構成を表すブロック図である。

### 【0164】

本実施形態においては、出力段選択回路 81、出力ライン選択回路 82、及び入力変数選択回路 2-4 を備えていることを特徴とする。尚、他の構成については、図 1 と同様であるため、説明は省略する。

### 【0165】

接続回路 5-1 ~ 5-3 及び論理関数メモリ 4-3 から出力される変数組は、出力段選択回路 81 において何れか一つが選択される。出力段選択回路 81 において選択された変数組は、入力変数 ( $F_1, F_2, F_3$ ) 及び選択切換変数 Select の値に基づいて、出力ライン選択回路 82 によりその総て又は一部が選択され、出力変数として出力される。

### 【0166】

ここで、出力変数の個数が 2 の冪乗である任意の  $n$  変数論理関数  $f(x_0, x_1, \dots, x_{n-1})$  は、以下の形で表現することが可能である。

### 【数 6】

$$\begin{aligned} & f(x_0, x_1, \dots, x_{n-3}, x_{n-2}, x_{n-1}) \\ &= x_{n-2}x_{n-1}f_0(x_0, x_1, \dots, x_{n-3}) \vee x_{n-2}\bar{x}_{n-1}f_1(x_0, x_1, \dots, x_{n-3}) \\ & \vee \bar{x}_{n-2}x_{n-1}f_2(x_0, x_1, \dots, x_{n-3}) \vee \bar{x}_{n-2}\bar{x}_{n-1}f_3(x_0, x_1, \dots, x_{n-3}) \end{aligned}$$

### 【0167】

従って、上記分解関数  $f_0 \sim f_3$  を論理関数メモリ 4-0 ~ 4-3 により実現し、各

分解関数 $f_0 \sim f_3$ と $x_{n-2}, x_{n-1}$ 等との積項を出力ライン選択回路 82 により演算することで、 $n$ 変数の論理関数の演算を行うことが可能となる。

#### 【0168】

出力段選択回路 81 は、マルチプレクサにより構成されており、演算制御部 10 から出力される出力選択信号  $t$  に従って、各論理関数メモリ 4-0 ~ 4-3 から出力される出力変数のうちの何れかを選択して出力する。

#### 【0169】

ここで、出力選択信号  $t$  とは、第 1 の出力選択回路 25 が選択する出力ライン束の指定番号 ( $t \in \{(00), (01), (10), (11)\}$ ) を表す 2 ビットの信号である。第 1 の出力選択回路 25 は、 $t = (00)$  のときは接続回路 5-1 の出力ライン束 (図 4 の  $o_8 \sim o_{15}$ ) を選択し、 $t = (01)$  のときは接続回路 5-2 の出力ライン束を選択し、 $t = (10)$  のときは接続回路 5-3 の出力ライン束を選択し、 $t = (11)$  のときは出力ライン選択回路 82 の出力ライン束を選択する。

出力ライン選択回路 82 は、入力変数選択回路 2-4 から入力される入力変数の値に従って、出力段選択回路 81 から出力される出力変数のうちの全部又は一部を選択して出力する。尚、入力変数選択回路 2-4 は、図 2 に示したものと同様のシフト回路で構成されている。但し、本実施形態では、最終段の論理関数メモリ 4-0 ~ 4-3 の出力が 8 ビットであるため、入力変数選択回路 2-4 に使用されるシフト回路の出力は、3 ビットとされる。一般には、入力変数選択回路 2-4 の出力は、最終段の論理関数メモリ 4-3 の出力の数  $N$  に対して、

#### 【数 7】

$$\lceil \log_2 N \rceil$$

とされる。以下、入力変数選択回路 2-4 の出力  $out(00) \sim out(02)$  (図 2 参照) を、それぞれ、 $F_0, F_1, F_2$  で表す。

#### 【0170】

図 23 は図 22 の出力ライン選択回路 82 の構成を表すブロック図である。尚、この回路図は、動作原理を説明するために、簡略化して記載している。

#### 【0171】

出力ライン選択回路 82 は、図 23 に示したように、2 入力 1 出力のマルチプレクサ（以下、「MUX」という。）83～89 を多段にカスケード状に接続し、各段において出力を、MUX90～MUX96 を通して取り出すことができるように構成されている。尚、図 23 では、説明の都合上、出力ラインのビット数  $N_c$  が 8 ビットの例を示しているが、 $N_c$  は 8 ビットに限られるものではない。

#### 【0172】

図 23 において、出力ライン選択回路 82 の入力ライン  $y_0 \sim y_7$  は、出力段選択回路 81 の出力側端子に接続される。この入力ライン  $y_0 \sim y_7$  より、出力変数  $Y$  の値が入力される。また、出力ライン選択回路 82 の出力ライン  $f^{(0)} \sim f^{(7)}$  からは、入力ライン  $y_0 \sim y_7$  のうち選択された出力変数  $Y$  の値 ( $y_0, \dots, y_{|Y|-1}$ ) が出力される。

#### 【0173】

入力ライン  $y_0, y_1$ 、入力ライン  $y_2, y_3$ 、入力ライン  $y_4, y_5$ 、及び入力ライン  $y_6, y_7$  は、それぞれ MUX83、MUX84、MUX85、及び MUX86 の入力側に接続されている。MUX83、84、及び MUX85、86 の出力は、それぞれ MUX87、及び MUX88 の入力側に接続されている。MUX87、88 の出力は、MUX89 の入力に接続されている。

#### 【0174】

一方、（入力ライン  $y_0$ 、MUX83 の出力）、（入力ライン  $y_2$ 、MUX84 の出力）、（入力ライン  $y_4$ 、MUX85 の出力）、及び（入力ライン  $y_6$ 、MUX86 の出力）は、それぞれ MUX90、MUX91、MUX92、及び MUX93 の入力側に接続されている。また、（入力ライン  $y_1$ 、MUX87 の出力）、（入力ライン  $y_3$ 、MUX88 の出力）、及び（入力ライン  $y_5$ 、MUX88 の出力）は、それぞれ MUX94、MUX95、及び MUX96 の入力側に接続されている。

#### 【0175】

MUX83～86 は、共通の入力変数  $F_0$  により切換制御がされる。すなわち、 $F_0$  が “0” のときは、MUX83、84、85、86 は、それぞれ、入力ライン  $y_0, y_2, y_4, y_6$  を選択し、 $F_0$  が “1” のときは、MUX83、84、85、86 は、それぞれ、入力ライン  $y_1, y_3, y_5, y_7$  を選択する。

## 【0176】

MUX 87, 88 は、共通の入力変数  $F_1$  により切換制御がされる。すなわち、 $F_1$  が “0” のときは、MUX 87, 88 は、それぞれ、MUX 83, 85 を選択し、 $F_1$  が “1” のときは、MUX 87, 88 は、それぞれ、MUX 84, 86 を選択する。

## 【0177】

また、MUX 89 は、入力変数  $F_2$  により切換制御がされる。すなわち、 $F_2$  が “0” のときは MUX 89 は、MUX 87 を選択し、 $F_2$  が “1” のときは、MUX 89 は、MUX 88 を選択する。

## 【0178】

MUX 83～86 によって、8 本の入力ライン  $y_0 \sim y_7$  のうちの 4 本が選択される。MUX 83～88 によって、8 本の入力ライン  $y_0 \sim y_7$  のうちの 2 本が選択される。更に、MUX 83～89 によって、8 本の入力ライン  $y_0 \sim y_7$  のうちの 1 本が選択される。これは、目的論理関数  $f$  の出力変数の個数  $|f|$  が 4 個以下の場合、出力段選択回路 81 の出力する中間変数  $Y$  に対して、更に入力変数 ( $F_0, F_1, F_2$ ) による論理演算を行うことが可能であることを意味する。従って、論理関数メモリ 4-0～4-3 により行う演算における入力変数の数を減らすことができる。

## 【0179】

すなわち、例えば、目的論理関数  $f(x_0, x_1, \dots, x_{n-2}, x_{n-1})$  を、

【数 8】

$$x_{n-1}f'(x_0, x_1, \dots, x_{n-2}) \vee \bar{x}_{n-1}f'(x_0, x_1, \dots, x_{n-2})$$

のようにシャノン展開して、関数  $f'(x_0, x_1, \dots, x_{n-2})$  を 4 つの分解関数  $f_0(X_0)$ ,  $f_1(X_1, Y_1)$ ,  $f_2(X_2, Y_2)$ ,  $f_3(X_3, Y_3)$  (但し、 $X_0 \cup X_1 \cup X_2 \cup X_3 = \{x_0, x_1, \dots, x_{n-2}\}$ ) に関数分解し、最終段の分解関数  $f_3(X_3, Y_3)$  の出力数を 4 以下となるようにすれば、論理関数メモリ 4-0～4-3 に入力する入力変数の個数を 1 個減らすことができる。そして、入力変数  $x_{n-1}$  は、入力変数  $F_0$  として、出力ライン選択回路 82 に入力するようにすればよい。

## 【0180】

同様に、目的論理関数  $f(x_0, x_1, \dots, x_{n-2}, x_{n-1})$  を、



## 【数 9】

$$x_{n-1}x_{n-2}f''(x_0, x_1, \dots, x_{n-3}) \vee x_{n-1}\bar{x}_{n-2}f''(x_0, x_1, \dots, x_{n-3}) \\ \vee \bar{x}_{n-1}x_{n-2}f''(x_0, x_1, \dots, x_{n-3}) \vee \bar{x}_{n-1}\bar{x}_{n-2}f''(x_0, x_1, \dots, x_{n-3})$$

のようにシャノン展開すれば、論理関数メモリ 4-0 ～ 4-3 に入力する入力変数の個数を 2 個減らすことができ、

## 【数 10】

$$x_{n-1}x_{n-2}x_{n-3}f'''(x_0, x_1, \dots, x_{n-4}) \vee x_{n-1}x_{n-2}\bar{x}_{n-3}f'''(x_0, x_1, \dots, x_{n-4}) \\ \vee x_{n-1}\bar{x}_{n-2}x_{n-3}f'''(x_0, x_1, \dots, x_{n-4}) \vee x_{n-1}\bar{x}_{n-2}\bar{x}_{n-3}f'''(x_0, x_1, \dots, x_{n-4}) \\ \vee \bar{x}_{n-1}x_{n-2}x_{n-3}f'''(x_0, x_1, \dots, x_{n-4}) \vee \bar{x}_{n-1}x_{n-2}\bar{x}_{n-3}f'''(x_0, x_1, \dots, x_{n-4}) \\ \vee \bar{x}_{n-1}\bar{x}_{n-2}x_{n-3}f'''(x_0, x_1, \dots, x_{n-4}) \vee \bar{x}_{n-1}\bar{x}_{n-2}\bar{x}_{n-3}f'''(x_0, x_1, \dots, x_{n-4})$$

のようにシャノン展開すれば、論理関数メモリ 4-0 ～ 4-3 に入力する入力変数の個数を 3 個減らすことができる。

## 【0181】

尚、MUX 83 ～ MUX 89 により選択された 8 ビット、4 ビット、2 ビット、又は 1 ビットの出力は、共通の 8 本の出力ラインによって出力するのが好ましい。ルックアップテーブル・カスケード論理回路を LSI により構成した場合、出力ライン数（ピン数）が少ないほどパッケージ面積を縮小できるからである。そこで、図 23 では、MUX 90 ～ 96 によって、上記各本数の出力ラインが選択されたときの出力を、共通の出力ライン  $f(0) \sim f(7)$  を通して出力されるように構成されている。MUX 90 ～ 96 は、共通の選択切換変数 Select によって切換制御がされる。尚、選択切換変数 Select の設定値は、出力変数の個数  $|f(X)|$  によって、あらかじめ選択切換メモリ 97 に記憶される。そして、演算制御部 10 の制御によって、Select に設定される。

## 【0182】

MUX 90 は、Select が “0” のとき入力ライン  $y_0$  を選択し、Select が “1” のとき MUX 83 の出力を選択し、選択したラインの信号値を出力ライン  $f(0)$  に出力する。MUX 91 は、Select が “0” のとき入力ライン  $y_2$  を選択し、Select が “1” のとき MUX 84 の出力を選択し、選択したラインの信号値を出力ライン  $f(2)$  に出力する。MUX 92 は、Select が “0” のとき入力ライン  $y_4$  を選択し、Select が

“1” のときMUX 8 5 の出力を選択し、選択したラインの信号値を出力ライン $f(4)$ に出力する。MUX 9 3 は、Selectが“0” のとき入力ライン $y_6$ を選択し、Selectが“1” のときMUX 8 6 の出力を選択し、選択したラインの信号値を出力ライン $f(6)$ に出力する。

#### 【0183】

MUX 9 4 は、Selectが“0” のとき入力ライン $y_1$ を選択し、Selectが“1” のときMUX 8 7 の出力を選択し、選択したラインの信号値を出力ライン $f(1)$ に出力する。MUX 9 5 は、Selectが“0” のとき入力ライン $y_3$ を選択し、Selectが“1” のときMUX 8 9 の出力を選択し、選択したラインの信号値を出力ライン $f(3)$ に出力する。MUX 9 6 は、Selectが“0” のとき入力ライン $y_5$ を選択し、Selectが“1” のときMUX 8 8 の出力を選択し、選択したラインの信号値を出力ライン $f(5)$ に出力する。また、出力ライン $f(7)$ は、入力ライン $y_7$ に直結されている。

#### 【0184】

目的論理関数 $f$ の演算結果 $f(X)$ のビット数（出力の本数）が8 のときは、Selectを“0” とする。これにより、出力ライン $f(0) \sim f(7)$ には、入力ライン $y_0 \sim y_7$ にされる出力変数 $Y=(y_0, \dots, y_7)$ の値が出力される。出力の本数が4 のときは、Selectを“1” に設定し、入力変数 $F_0$ に“0” 又は“1” をする。これにより、入力ライン $y_0 \sim y_7$ のうち $F_0=0$ のときは $(y_0, y_2, y_4, y_6)$ の4 本が、 $F_0=1$ のときは $(y_1, y_3, y_5, y_7)$ の4 本が、出力ライン $f(0), f(2), f(4), f(6)$ に出力される。出力の本数が2 のときは、Selectを“1” とし、入力変数 $F_0, F_1$ に“0” 又は“1” を設定する。これにより、 $(F_0, F_1)=(0, 0)$ のときは、出力ライン $f(1), f(5)$ には入力ライン $y_0 \sim y_7$ のうち2 本にされる出力変数 $Y=(y_0, y_4)$ の値が出力される。 $(F_0, F_1)=(0, 1), (1, 1)$ のときは出力には $(y_1, y_5), (y_2, y_6), (y_3, y_7)$ の値がそれぞれ出力される。出力の本数が1 のときは、Selectを“1” とし、入力変数 $F_0, F_1, F_2$ に“0” 又は“1” を設定する。これにより、出力ライン $f(3)$ には、入力ライン $y_0 \sim y_7$ のうち1 本に $F_0, F_1, F_2$ で指定したが出力される。

#### 【0185】

このように出力ライン選択回路8 2 を設けることにより、各論理関数メモリ4

-0～4-3より出力される出力変数の個数が、全出力ラインの本数の $1/2$ 以下の場合には、出力ライン選択回路82を用いて更に論理演算を行うことが可能である。これによって、論理関数メモリ4-0～4-3において演算を行う分解関数の入力変数の総数を1個以上減らすことが可能となる。従って、ルックアップテーブル・カスケード論理回路全体で許容される入力変数の数を増やすことが可能となる。

#### 【0186】

尚、図23では、出力ライン選択回路82内のマルチプレクサは2入力1出力のものを使用しているが、一般に、 $w$ 入力1出力( $w \geq 2$ )のマルチプレクサを使用することが可能である。

#### 【0187】

##### 【発明の効果】

以上のように本発明の第1の構成によれば、目的関数に適合させて接続関係を自由に変更することができるため、論理回路をプログラマブルに構成することが可能となる。そして、目的関数の入力変数の数が各論理関数メモリの入力端子数の和よりも多い場合であっても、フィードバック回路によって最後段の論理関数メモリの出力を最前段の論理関数メモリにフィードバックさせて、同じ論理関数メモリを重複して使用して分解関数の演算を行わせることが可能となる。すなわち、演算が可能な目的関数の入力変数の個数の自由度が非常に大きく、設計自由度の大きいルックアップテーブル・カスケード論理回路を提供することができる。

#### 【0188】

本発明の第2の構成によれば、最後段の論理関数メモリから出力される中間変数の選択及び最前段の論理関数メモリとの接続関係を、フィードバック用接続回路とフィードバック用接続メモリによってプログラマブルに設定することができるため、フィードバック後に最前段の論理関数メモリで演算を行う分解関数のルックアップテーブルの構成の自由度が高いルックアップテーブル・カスケード論理回路を提供することができる。また、フィードバックの際の最前段の論理関数メモリの入出力端子を有効に利用することが可能となるとともに、最前段の論理

関数メモリのメモリ容量を効率よく活用することが可能となるため、演算が可能な目的関数の自由度の高いルックアップテーブル・カスケード論理回路を小容量の論理関数メモリで実現することが可能となる。

#### 【0189】

本発明の第3の構成によれば、ルックアップテーブル・カスケード論理回路をパイプライン処理により非同期的に動作させる場合であっても、各論理関数メモリに記憶された分解関数の形によらず、各論理関数メモリの出力が発振により不安定化することを防止することが可能となる。

#### 【0190】

本発明の第4の構成によれば、ページ切換手段によって論理関数メモリ又は接続メモリ若しくはフィードバック用接続メモリ内の入出力が可能なページを選択的に切り換えることで、1つの論理関数メモリで複数の種類の分解関数の演算を行うことが可能となる。従って、複数の論理関数の演算を1個のルックアップテーブル・カスケード論理回路によって行うことが可能となる。また、最後段の論理関数メモリの出力を最前段の論理関数メモリの入力にフィードバックさせて使用する場合、各演算のループによって論理関数メモリ若しくは接続メモリ又はフィードバック用接続メモリのページを切り換えて使用することで、ルックアップテーブル・カスケード論理回路によって演算することが可能な論理関数の自由度を広げることができる。

#### 【0191】

本発明の第5の構成によれば、簡単な回路により接続回路を実現することが可能となる。

#### 【0192】

本発明の第6の構成によれば、接続回路が、前段の論理関数メモリの出力を、論理関数の演算結果を出力する出力ラインに接続可能としたことで、目的論理関数の分解関数の段数が少ない場合には、カスケードの途中の論理関数メモリの出力を出力変数として取り出し、演算速度を高速化することが可能となる。

#### 【0193】

本発明の第7の構成によれば、各論理関数メモリから出力される出力を、出力

段選択回路により自由に選択して出力させることができるので、外部への出力ラインの本数を減らすことが可能となる。従って、集積回路で構成する場合は、出力変数を出力するための集積回路のピン数を削減することが可能となり、小型化することができる。

#### 【0194】

本発明の第8の構成によれば、第1の選択された段の出力変数を、第1の選択された段の分解関数の出力変数の個数が、第1の選択された段の論理関数メモリの出力数の $1/2$ 以下の場合、入力変数のうちの一部のものの値に基づいて、第1の選択された段の分解関数の出力変数を選択することが可能となり、出力ライン選択回路において更に論理演算を行うことができる。従って、各論理関数メモリに入力する入力変数の個数を削減することができる。そのため、論理関数メモリのメモリ使用効率が向上する。また、ルックアップテーブル・カスケード論理回路全体で演算が可能な論理関数の入力変数の個数に関する制限数を広げることが可能となる。

#### 【0195】

本発明の第9の構成によれば、前段又は最終段の論理関数メモリから出力される中間変数の個数と、各段の論理関数メモリに入力する入力変数の個数とを、接続メモリ及び接続回路又はフィードバック用接続メモリ及びフィードバック用接続回路を用いて、各分解関数に適合させて変更することが可能となるため、各論理関数メモリのメモリ容量を有効に活用することができる。

#### 【0196】

本発明の第10の構成によれば、最初の段以外の各論理関数メモリの入力のうちの少なくとも1つに、接続回路を介することなく入力変数を直接入力することにより、接続回路の入力ラインの数を減らすことができる。そのため、接続回路及び接続メモリの出力の配線数を減らすことができ、回路の高集積化及び省電力化が可能となる。

#### 【0197】

本発明の第11の構成によれば、論理関数メモリ又は接続回路若しくはフィードバック用接続回路に入力される入力変数の個数を減らすことができるため、接

続回路及び接続メモリ並びにフィードバック用接続回路及びフィードバック用接続メモリの出力の配線数を減らすことができ、回路を高集積化及び省電力化が可能となる。

#### 【0198】

本発明の第12の構成によれば、簡易な回路によって入力選択回路を構成することができ、また、入力選択回路における入力変数の選択も入力変数のシフトビット数を指定するだけでよいため、入力選択回路の制御ラインが少なくなる。従って、スイッチング素子の数や配線面積を減らすことができるため、高集積化及び省電力化を行うことが可能となる。

#### 【0199】

本発明の第13の構成によれば、各論理関数メモリの演算を同時に行うことが可能であり、各論理関数メモリで並行して演算処理を行うことが可能となるため、演算速度を向上させることが可能となる。

#### 【0200】

本発明の第14の構成によれば、最初の段以外の各論理関数メモリの入力の一部を、接続回路を介することなく前段の論理関数メモリの出力を直接入力することにより、接続回路の入力ラインの数を減らすことができる。これにより、接続回路及び接続メモリの素子数及び出力の配線数を減らすことができるため、回路を高集積化・省電力化することが可能となる。

#### 【図面の簡単な説明】

【図1】 本発明の実施形態1に係るルックアップテーブル・カスケード論理回路の全体構成を表すブロック図である。

【図2】 図1の入力変数選択回路2-i ( $i \in \{0, 1, 2, 3\}$ )の構成を表す回路図である。

【図3】 図1の論理関数メモリ4の構成を表すブロック図である。

【図4】 図1の接続回路5の周辺の回路ブロック図である。

【図5】 図1の接続メモリ6の構成を表すブロック図である。

【図6】 図1の演算制御部10の構成を表すブロック図である。

【図7】 図6の出力コントローラ35の内部構成を表すブロック図である。

。 【図 8】 実施形態 1 に係るルックアップテーブル・カスケード論理回路の動作を表すフローチャートである。

【図 9】 実施形態 1 に係るルックアップテーブル・カスケード論理回路の動作時における各信号の変化を表すタイミング図である。

【図 10】 図 1 の接続回路をクロスバスイッチにより構成した例を表す図である。

【図 11】 図 1 の接続回路をマルチプレクサにより構成した例を表す図である。

【図 12】  $2n$  ビットの二進数  $A, B$  の加算を行う論理関数  $f$  を表す図である。

【図 13】  $2n$  ビットの二進数  $A, B$  の加算を行う論理関数  $f$  を  $2n$  個の分解関数  $\{g_0, g_1, \dots, g_{2n-1}\}$  に関数分解した図である。

【図 14】  $2n$  ビットの二進数  $A, B$  の加算を行う論理関数  $f$  を  $2n$  個の分解関数  $\{g_0, g_1, \dots, g_{2n-1}\}$  に関数分解した図である。

【図 15】  $2n$  ビットの二進数  $A, B$  の加算を行う論理関数  $f$  を  $n$  個の分解関数  $\{f_0, f_1, \dots, f_{n-1}\}$  に関数分解した図である。

【図 16】 分解関数  $f_i$  の真理値表である。

【図 17】 本発明の実施形態 2 に係るルックアップテーブル・カスケード論理回路の全体構成を表すブロック図である。

【図 18】 本発明の実施形態 3 に係るルックアップテーブル・カスケード論理回路の全体構成を表すブロック図である。

【図 19】 本発明の実施形態 4 に係るルックアップテーブル・カスケード論理回路の全体構成を表すブロック図である。

【図 20】 図 19 の出力レジスタ及び出力デコーダの構成を表すブロック図である。

【図 21】 図 20 の記憶素子の構成を表すブロック図である。

【図 22】 本発明の実施形態 5 に係るルックアップテーブル・カスケード論理回路の全体構成を表すブロック図である。

【図 2 3】 図 2 2 の出力ライン選択回路 8 2 の構成を表すブロック図である。

【図 2 4】 ルックアップテーブル・カスケード論理回路の原理を表す図である。

【符号の説明】

- 1 入力変数レジスタ
- 2-0 ~ 2-4, 60-0 ~ 60-3 入力変数選択回路
- 3-0 ~ 3-4, 61-1 ~ 61-3 入力選択メモリ
- 4-0 ~ 4-3 論理関数メモリ
- 5-0 ~ 5-3 接続回路
- 6-0 ~ 6-3 接続メモリ
- 8-0 ~ 8-3 ページ選択メモリ
- 10 演算制御部
- 11-0, 20-0 1ビットシフト回路
- 11-1, 20-1 2ビットシフト回路
- 11-2, 20-2 4ビットシフト回路
- 11-3, 20-3 8ビットシフト回路
- $FM_0 \sim FM_{m-1}$  メモリアレイ
- 15, 21 ページデコーダ
- 16 アドレスデコーダ
- 17 入力ライン
- 18 出力ライン
- 31 演算ステップレジスタ
- 32 ステップカウンタ
- 33 ループカウンタ
- 34 ページカウンタ
- 35 出力コントローラ
- 41 ~ 44 フリップ・フロップ (FF)
- 45 ~ 48 AND回路

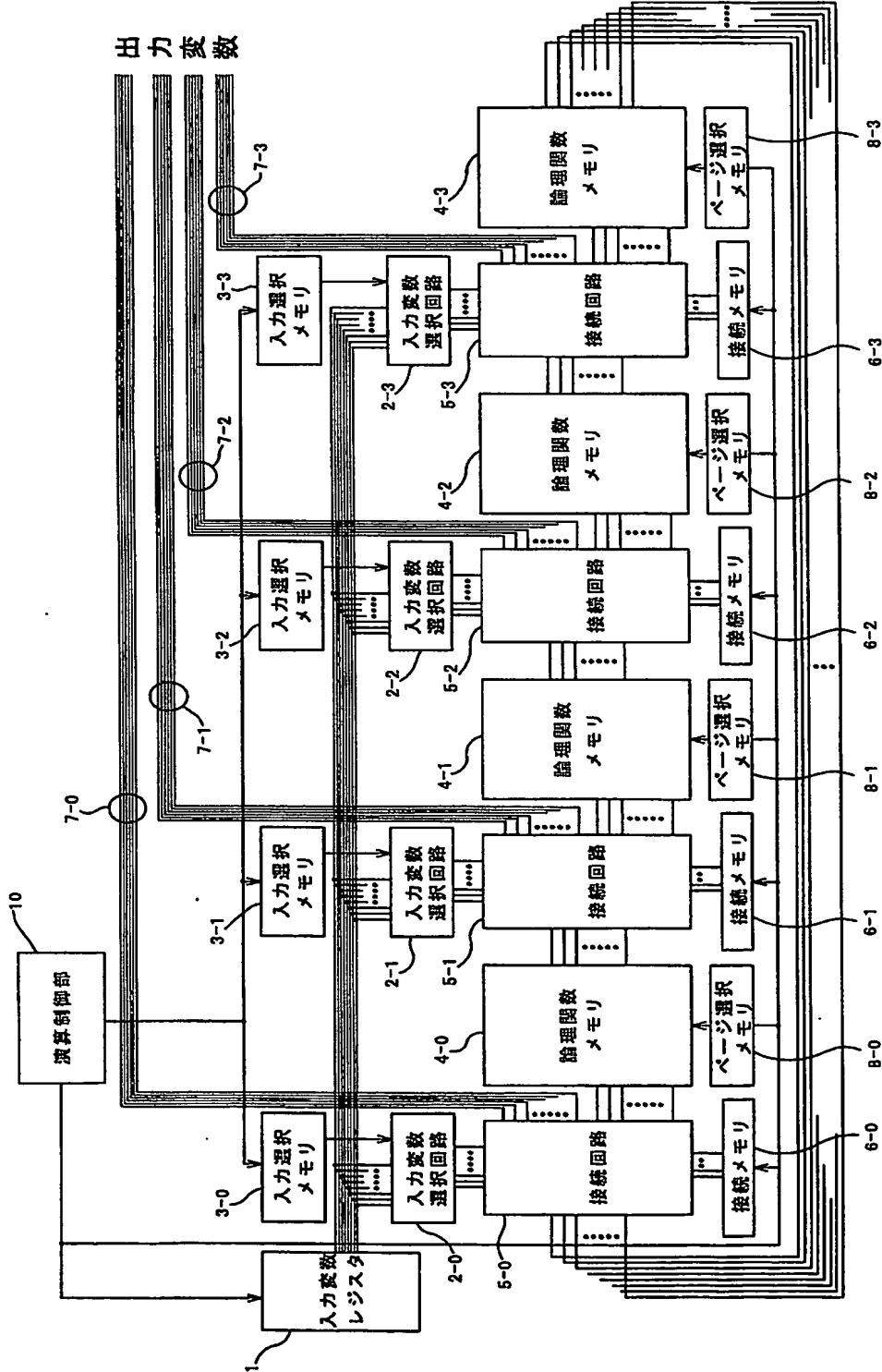


- 4 9 OR回路
- 6 5 中間変数バッファ
- 7 1 出力変数レジスタ
- 7 2 出力選択デコーダ
- 7 7 Dフリップ・フロップ
- 7 8, 8 3 ~ 9 6 マルチプレクサ (MUX)
- 8 1 出力段選択回路
- 8 2 出力ライン選択回路
- 9 7 選択切換メモリ

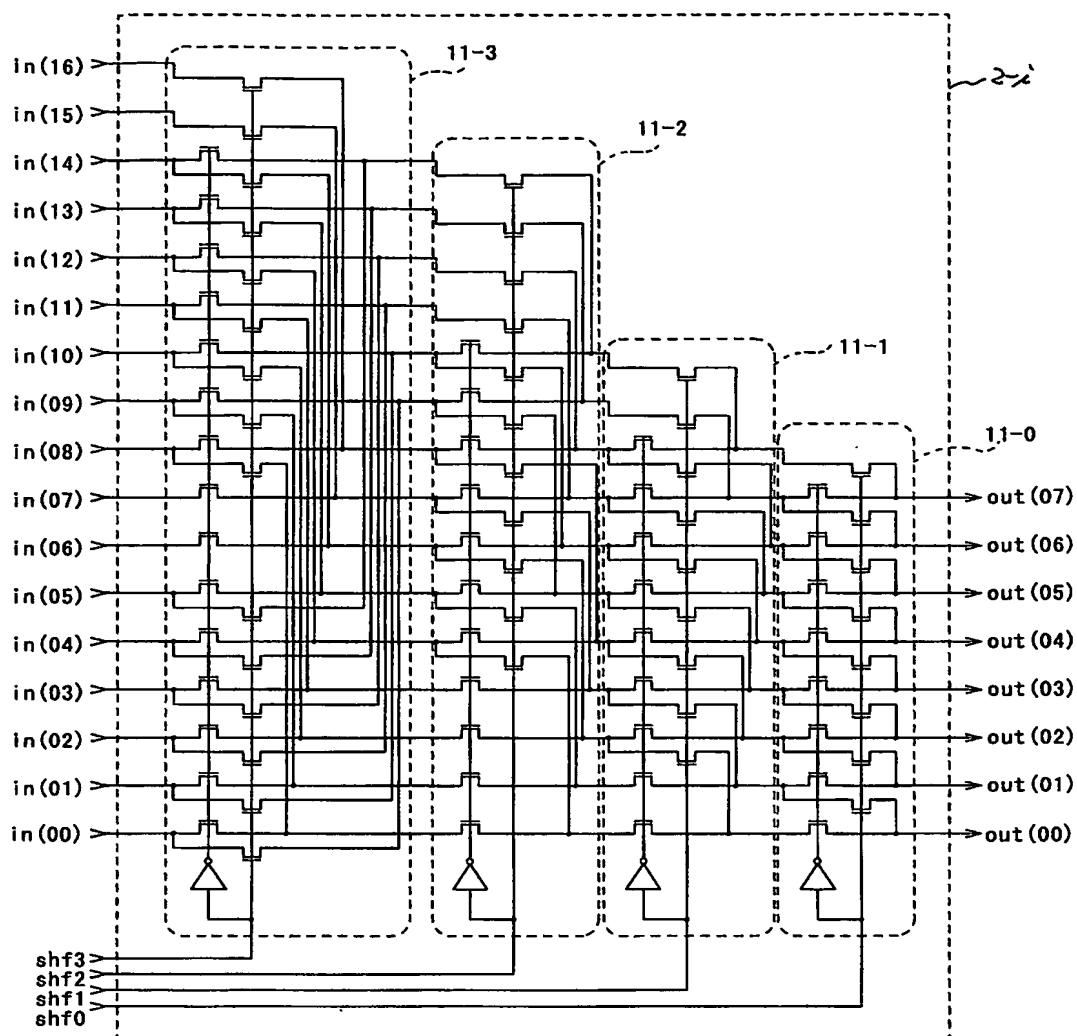
【書類名】

図面

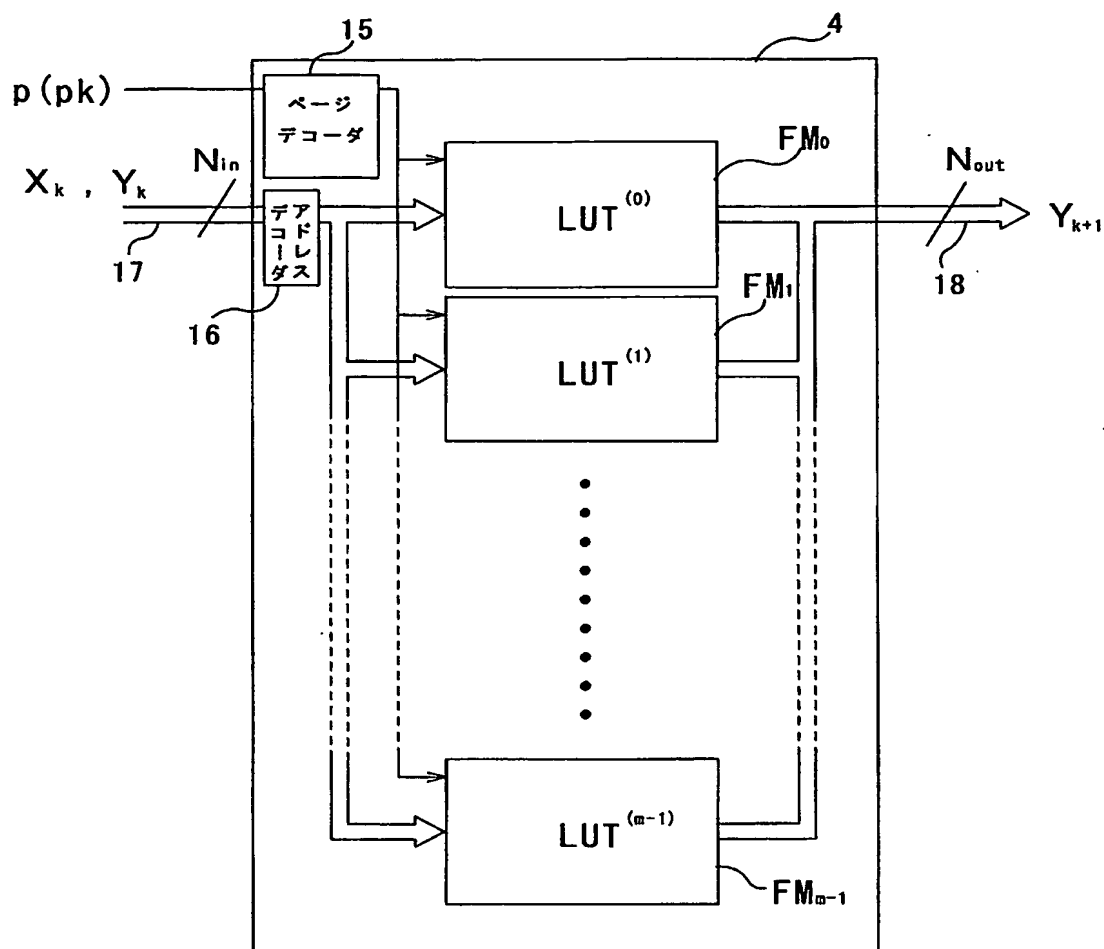
【図1】



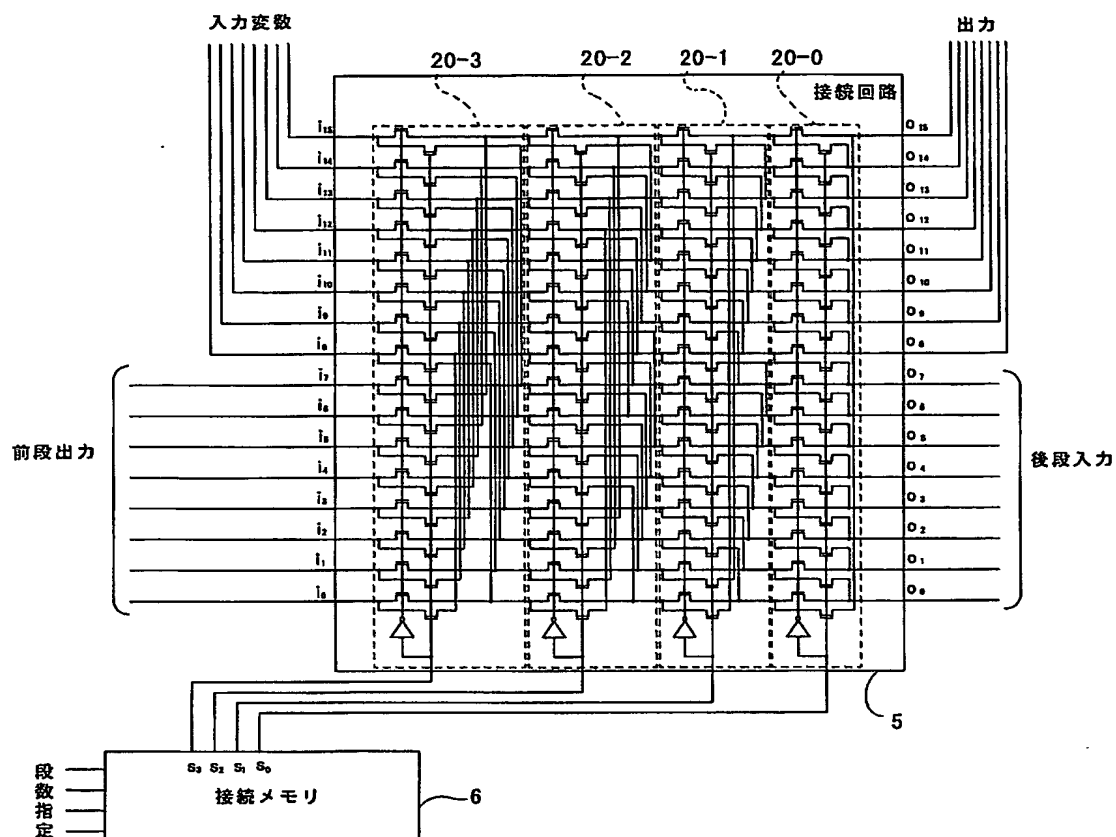
【図 2】



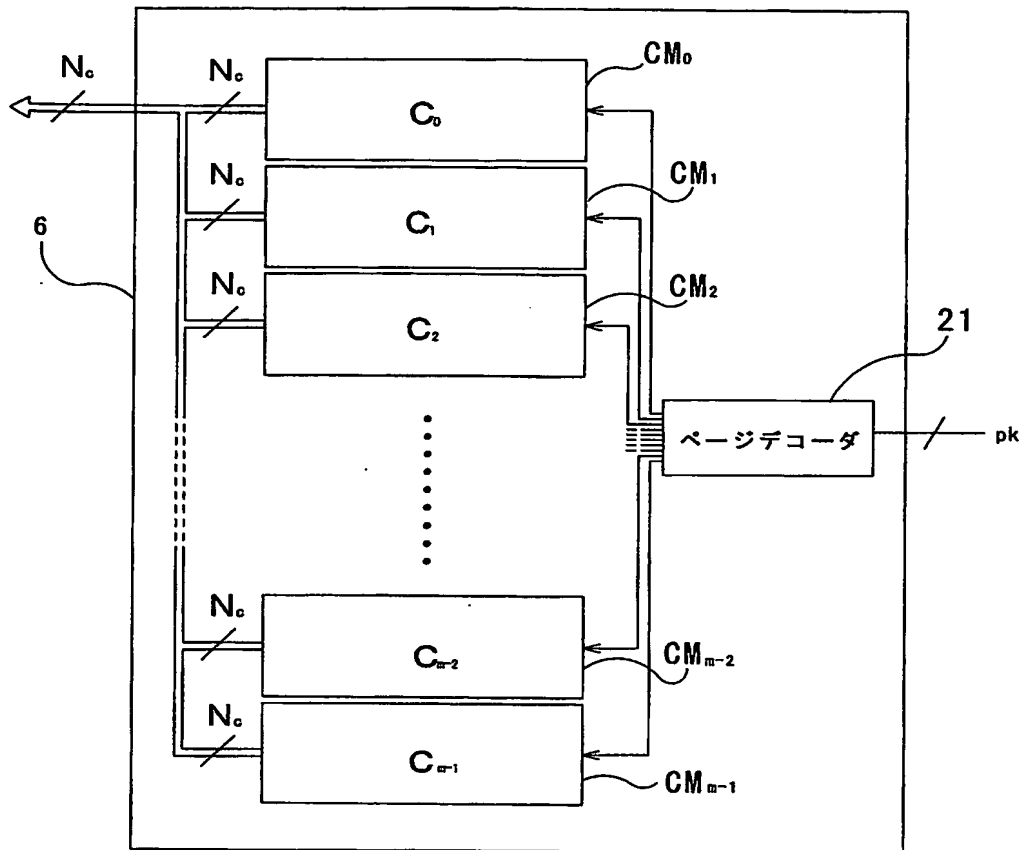
【図 3】



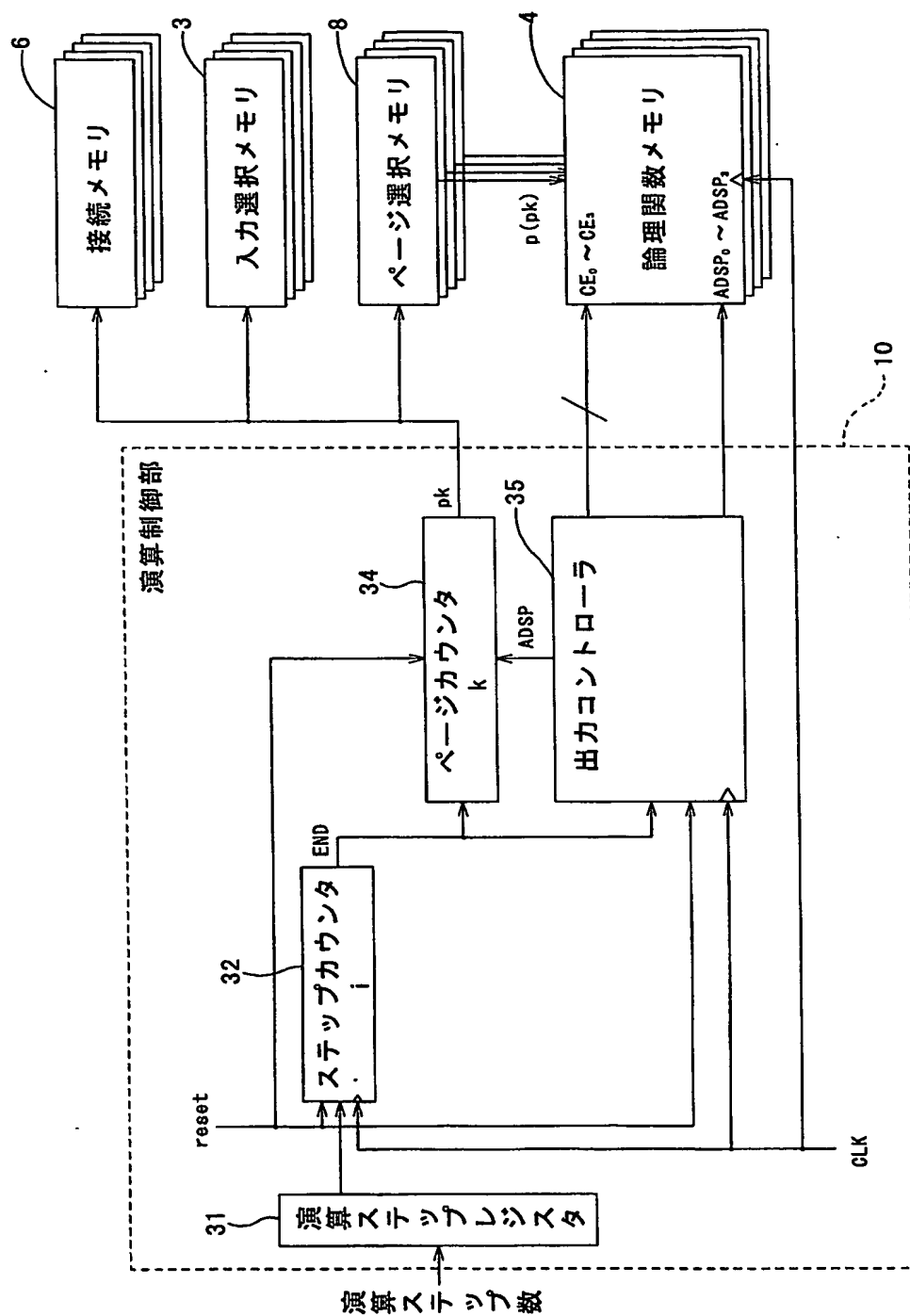
【図 4】



【図 5】



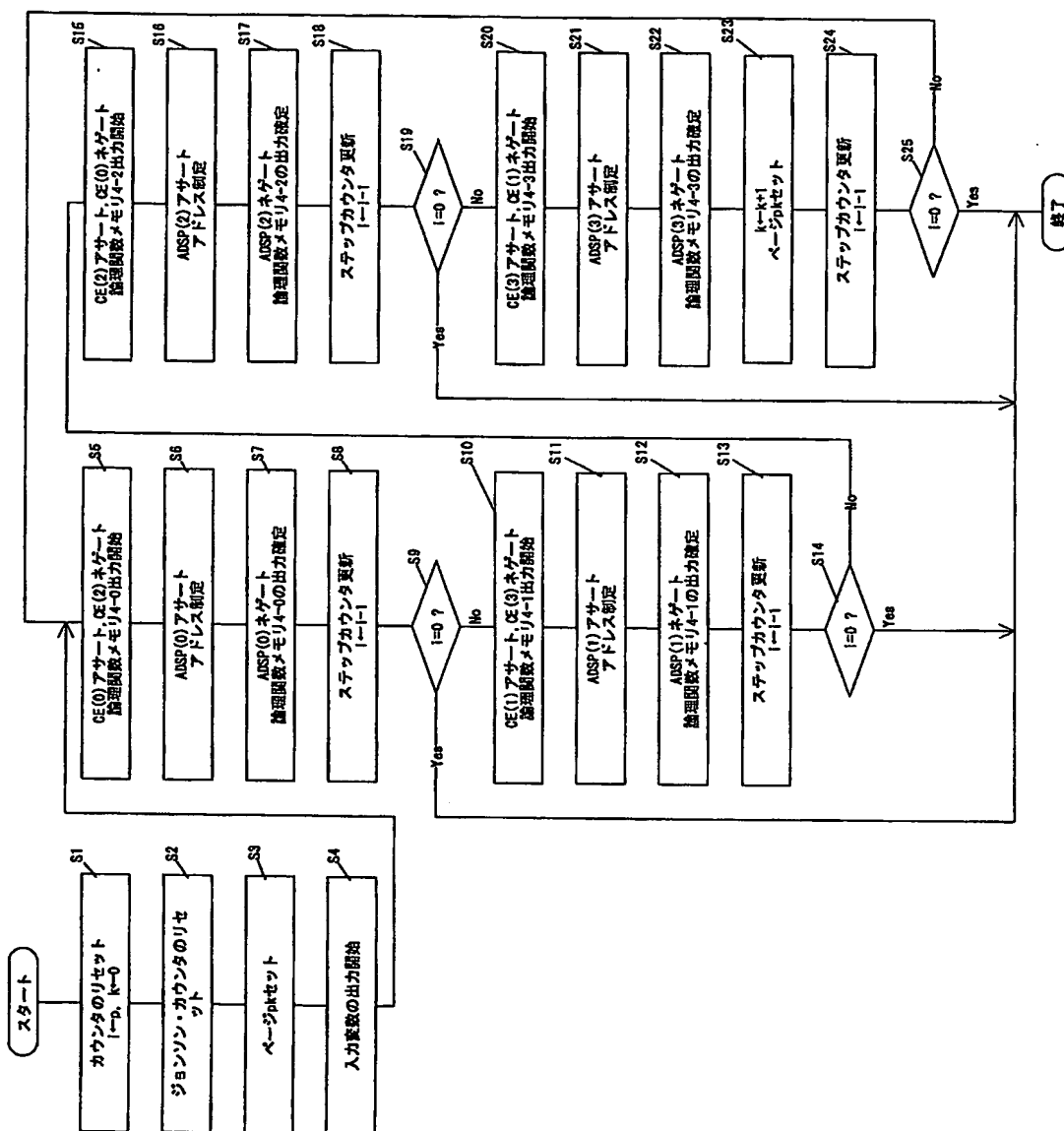
【図6】



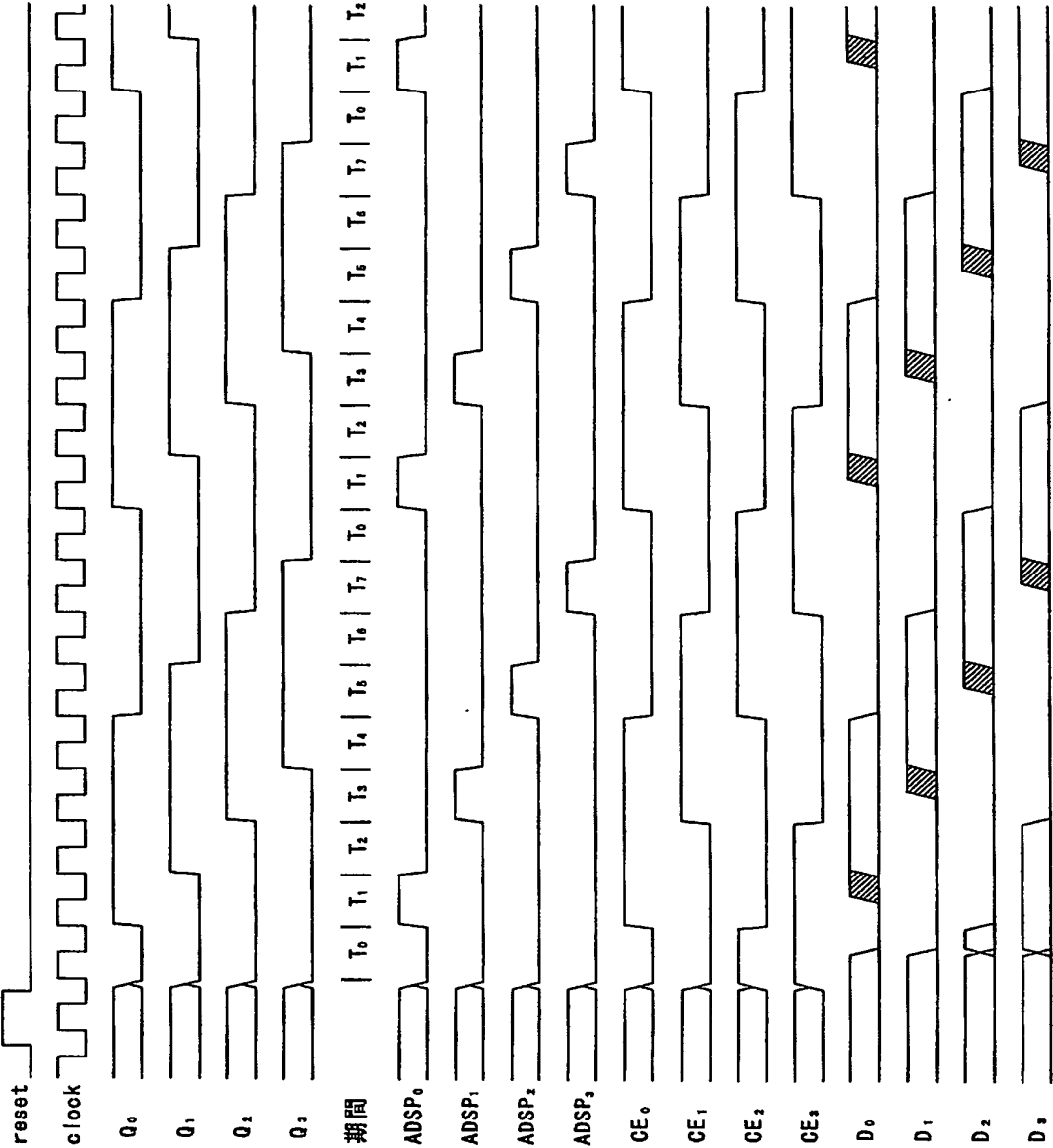




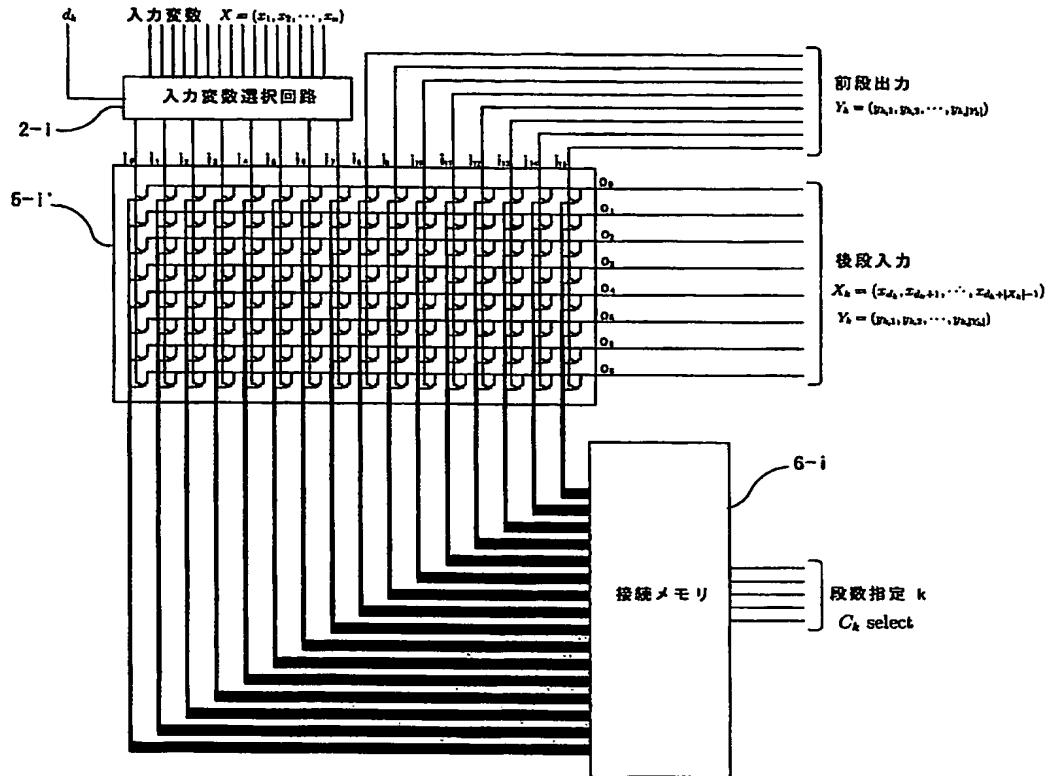
【图 8】



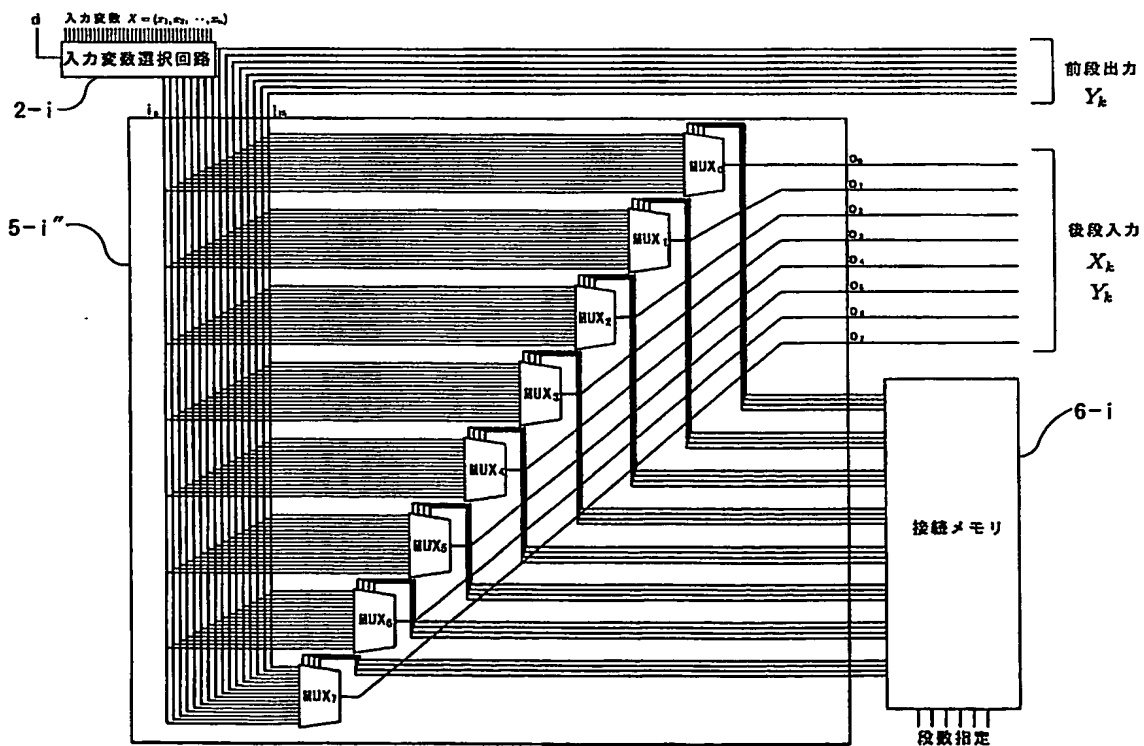
【図 9】



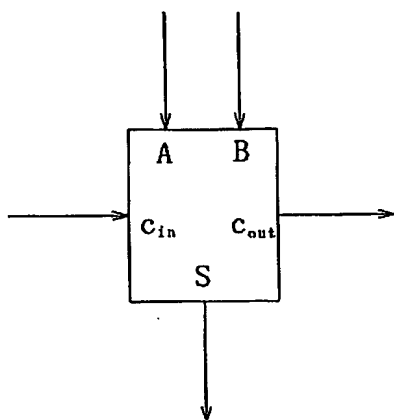
【図 10】



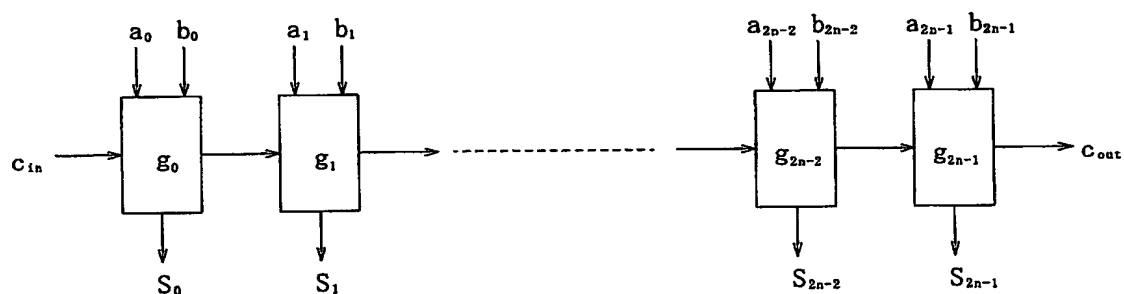
【図 11】



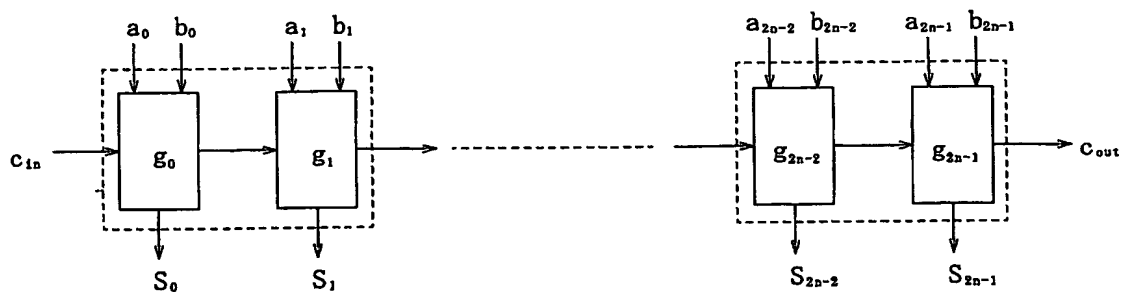
【図 12】



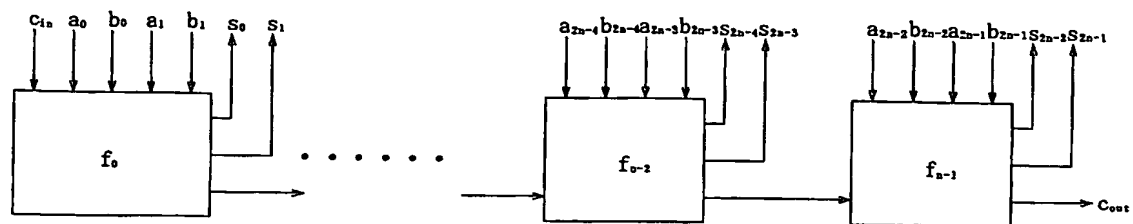
【図 13】



【図 14】



【図 15】



【図 16】

(a)

$c_{in}$	$a_1$	$a_0$	$b_1$	$b_0$	$c_{out}$	$s_1$	$s_0$
0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	1
0	0	0	1	0	0	1	0
0	0	0	1	1	0	1	1
0	0	1	0	0	0	0	1
0	0	1	0	1	0	1	0
0	0	1	1	0	0	1	1
0	0	1	1	1	1	0	0
0	1	0	0	0	0	1	0
0	1	0	0	1	0	1	1
0	1	0	1	0	1	0	0
0	1	0	1	1	1	0	1
0	1	1	0	0	0	1	1
0	1	1	0	1	1	0	0
0	1	1	1	0	1	0	1
0	1	1	1	1	1	1	0

$c_{in}$	$a_1$	$a_0$	$b_1$	$b_0$	$c_{out}$	$s_1$	$s_0$
1	0	0	0	0	0	0	1
1	0	0	0	1	0	1	0
1	0	0	1	0	0	1	1
1	0	0	1	1	1	0	0
1	0	1	0	0	0	1	0
1	0	1	0	1	0	1	1
1	0	1	1	0	1	0	0
1	0	1	1	1	1	1	0
1	1	0	0	0	0	1	1
1	1	0	0	1	1	0	0
1	1	0	1	0	1	0	1
1	1	0	1	1	1	1	0
1	1	1	0	0	1	0	0
1	1	1	0	1	1	0	1
1	1	1	1	0	1	1	0
1	1	1	1	1	1	1	1

(b)

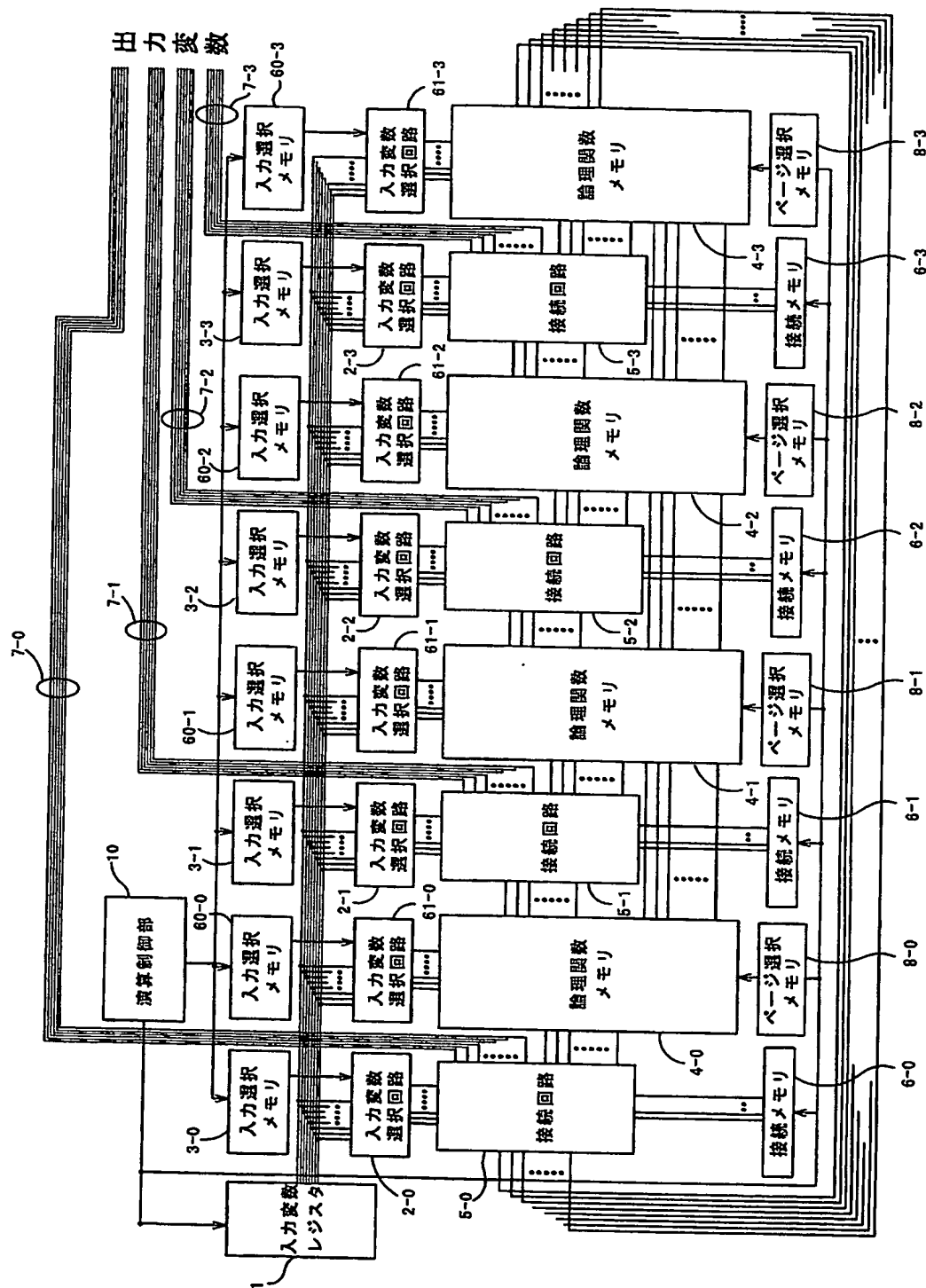
アドレス								出力値							
$i_0$	$i_1$	$i_2$	$i_3$	$i_4$	$i_5$	$i_6$	$i_7$	$o_7$	$o_6$	$o_5$	$o_4$	$o_3$	$o_2$	$o_1$	$o_0$
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0
0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0
0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0
0	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0
0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0
0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0
0	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0

アドレス								出力値							
$i_0$	$i_1$	$i_2$	$i_3$	$i_4$	$i_5$	$i_6$	$i_7$	$o_7$	$o_6$	$o_5$	$o_4$	$o_3$	$o_2$	$o_1$	$o_0$
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0
1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
1	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0
1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0
1	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0
1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0
1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0
1	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0
1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0
1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0
1	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0
1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0
1	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0

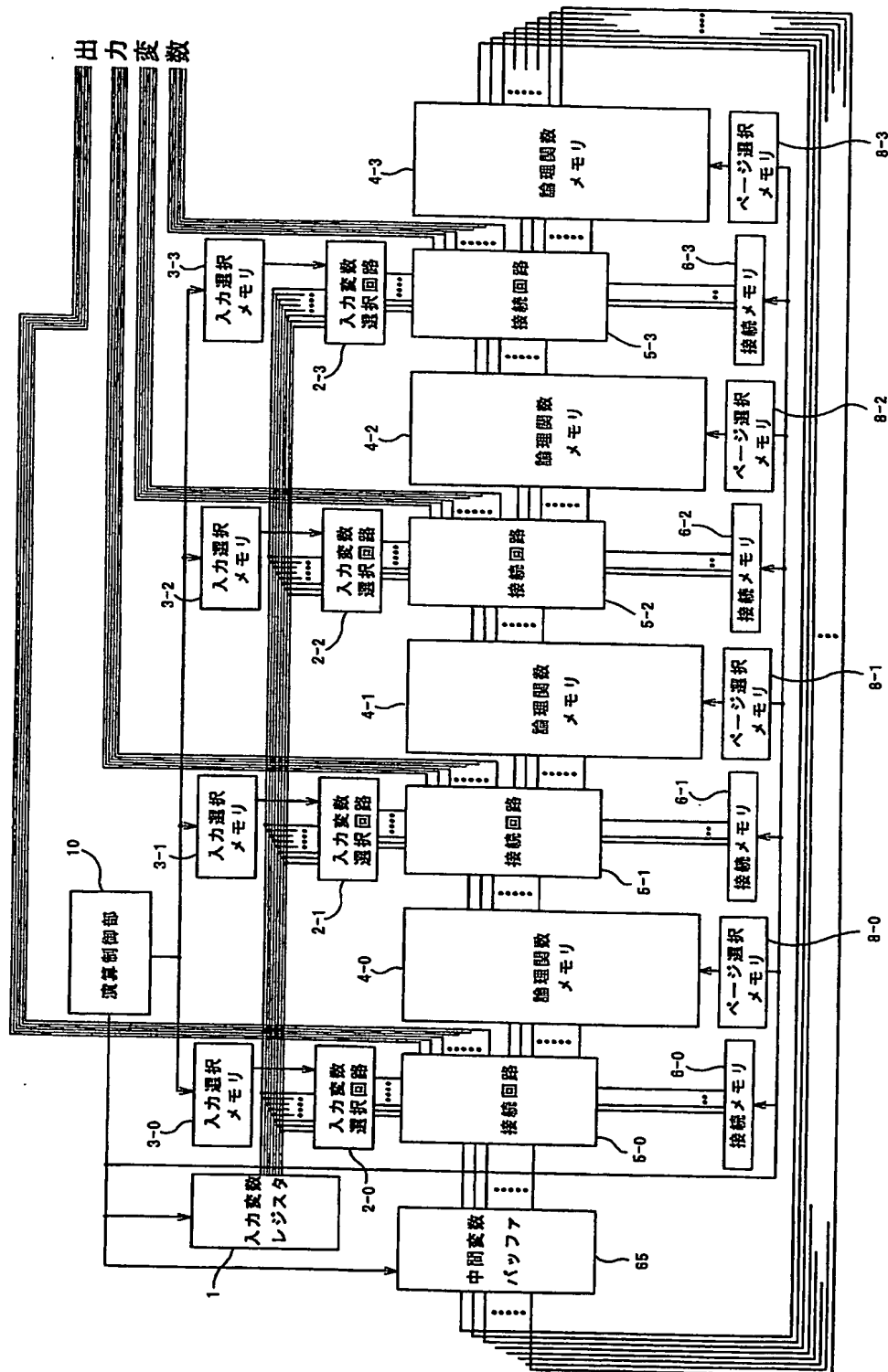
(c)

アドレス								出力値							
$i_0$	$i_1$	$i_2$	$i_3$	$i_4$	$i_5$	$i_6$	$i_7$	$o_7$	$o_6$	$o_5$	$o_4$	$o_3$	$o_2$	$o_1$	$o_0$
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0
0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0
0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0
0	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0
0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0
0	1	1	0	1	0	0	0	0							

【図 17】

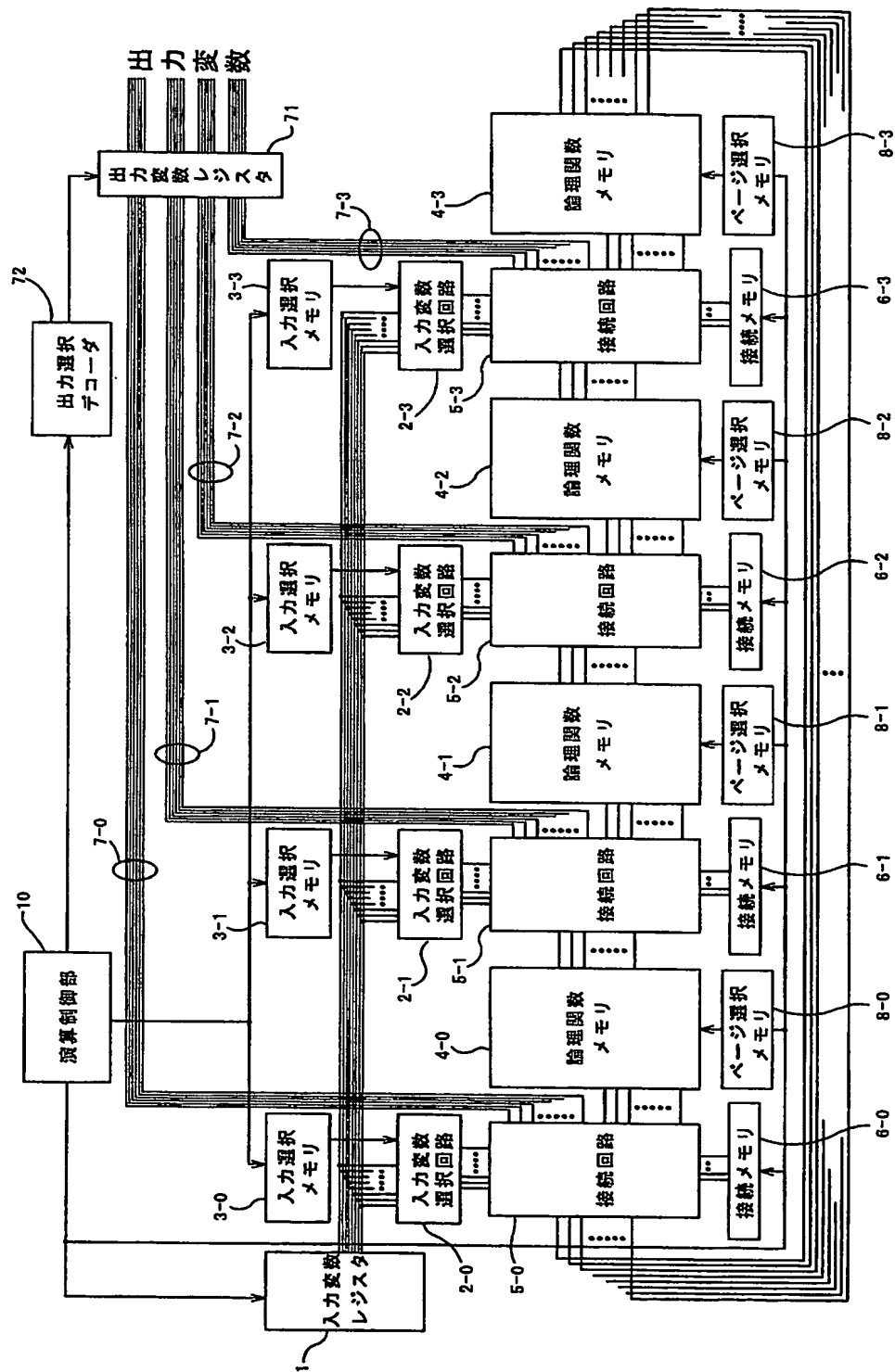


【図 18】

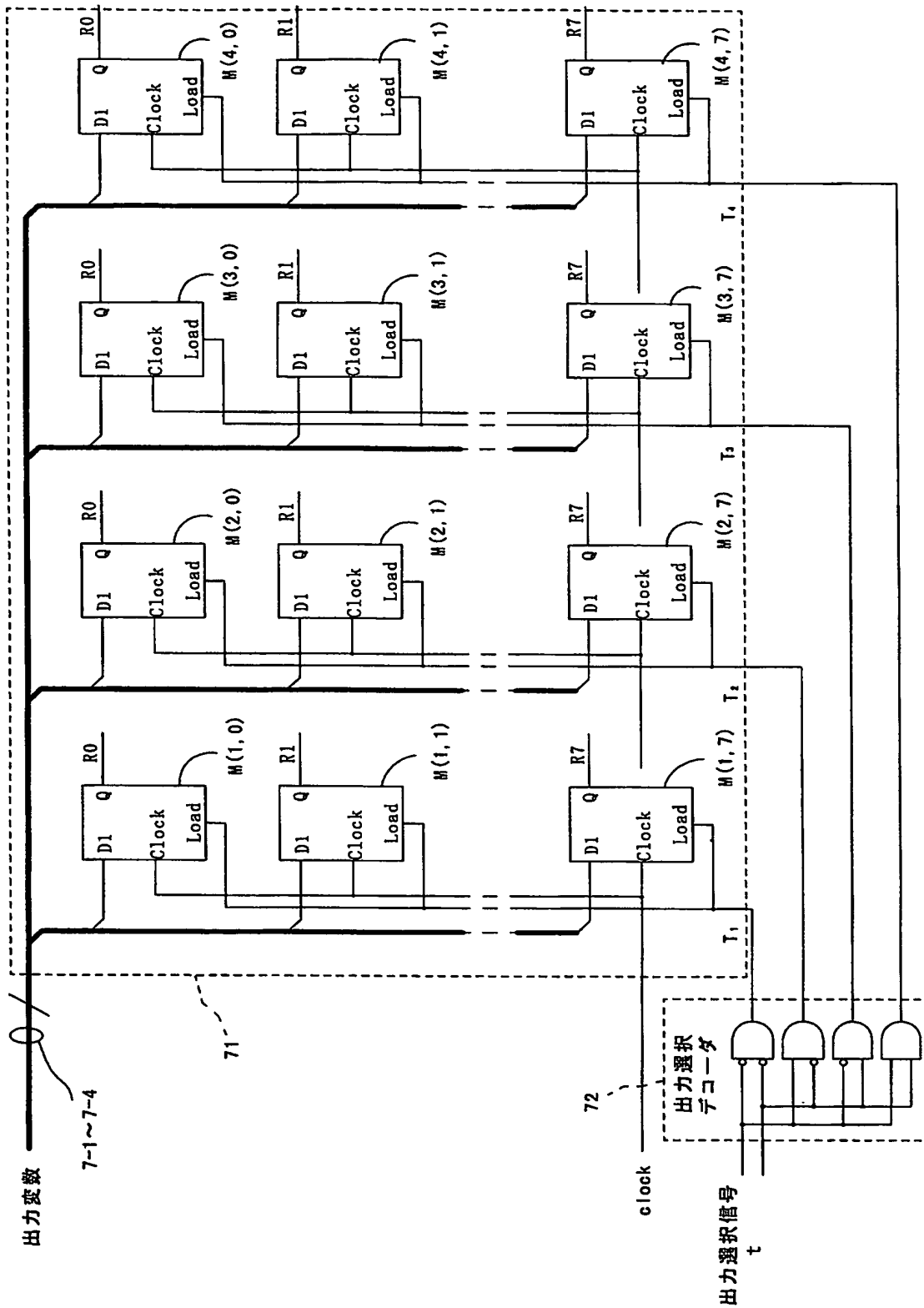




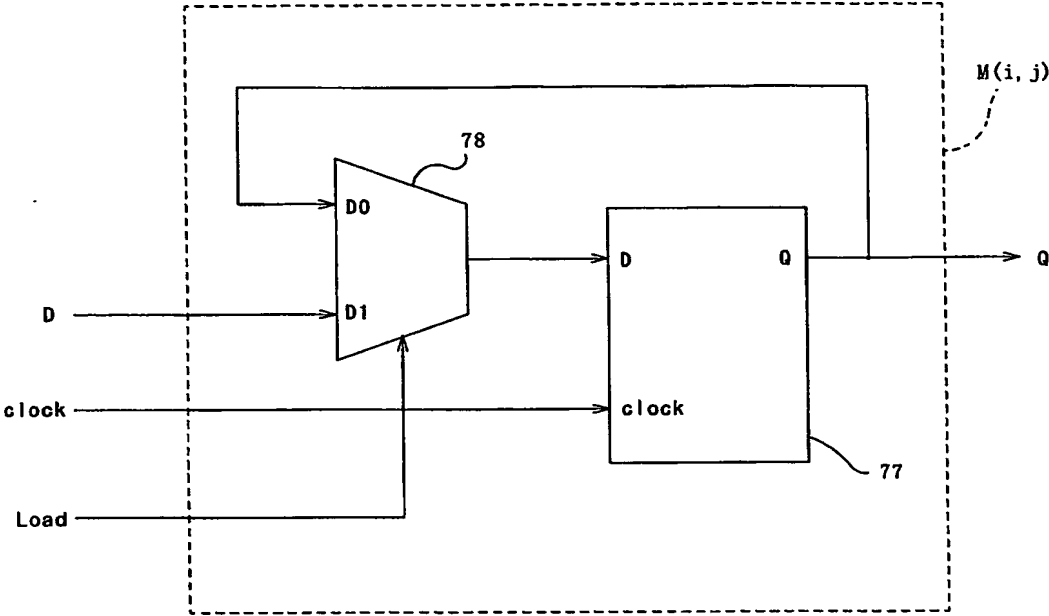
【図 19】



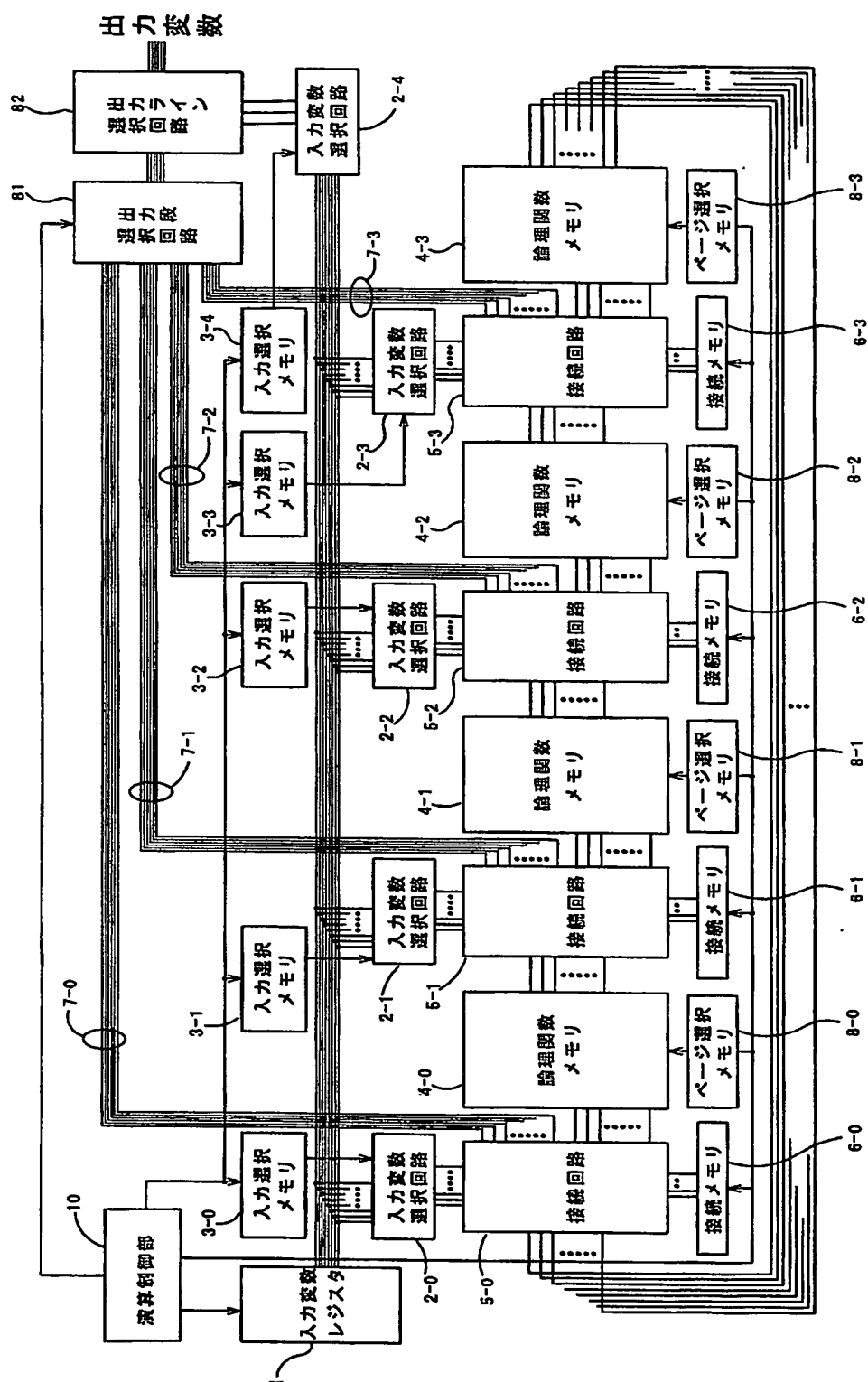
【図20】



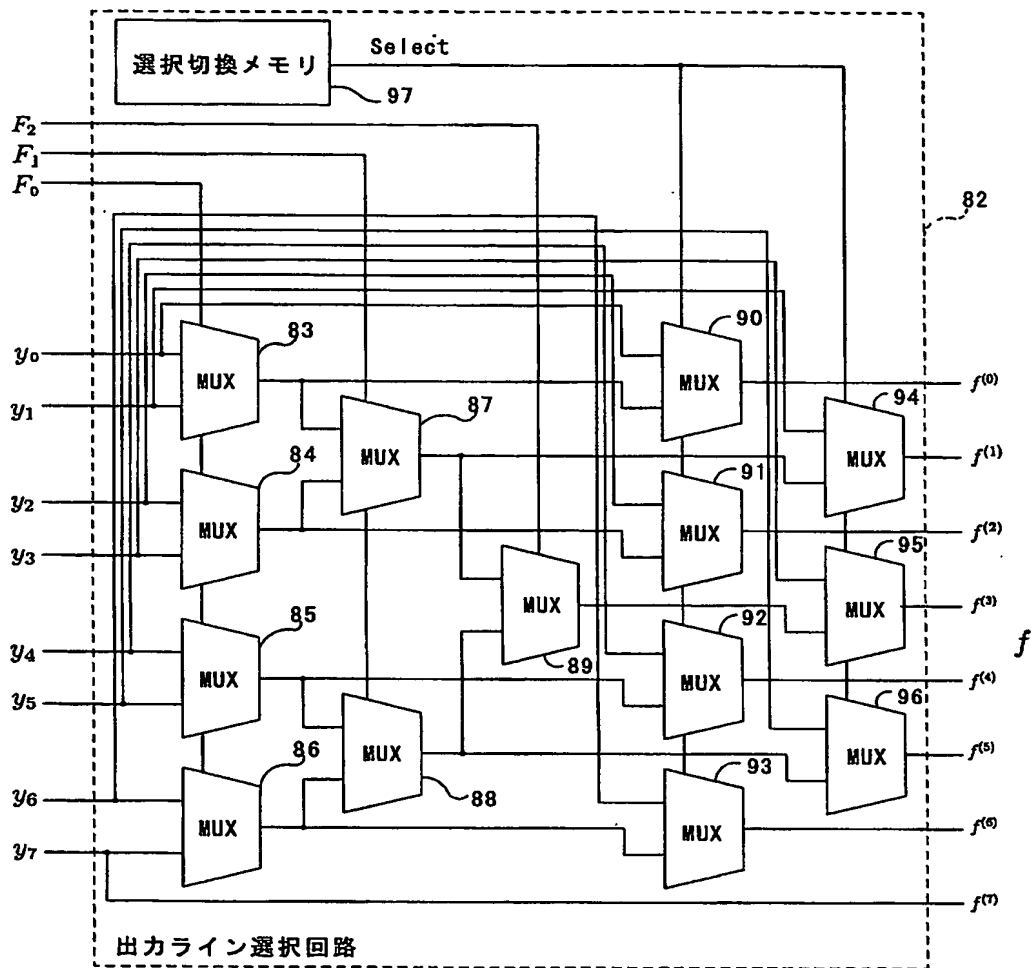
【図 21】



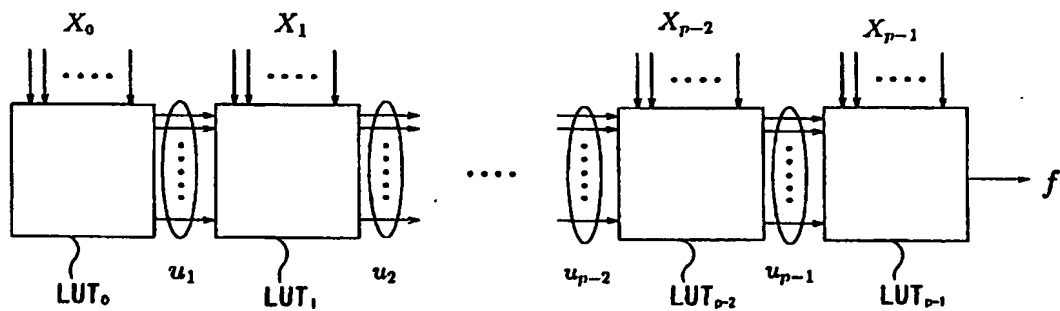
【図 22】



【図 2 3】



【図 2 4】



【書類名】 要約書

【要約】

【課題】 目的論理関数に応じ入力変数・レイル数を柔軟に変化でき、分解関数の数に応じてカスケード段数を柔軟に変化できるルックアップテーブル・カスケード論理回路を提供する。

【解決手段】 各分解関数をルックアップテーブルとして記憶する、直列に配列された複数の論理関数メモリ 4-0～4-3、前段の論理関数メモリの出力のうち、後段の論理関数メモリの入力に接続されるものの接続情報を記憶する接続メモリ 6-0～6-3、接続メモリより出力される接続順序に従い、前段の論理関数メモリの出力と後段の論理関数メモリの入力を選択的に接続する接続回路 5-1～5-3、及び最後段の論理関数メモリの出力を最前段の論理関数メモリに選択的に入力するフィードバック回路 5-0， 6-0 を備えた構成とした。

【選択図】 図 1

認定・付加情報

特許出願の番号	特願 2003-105762
受付番号	50300590290
書類名	特許願
担当官	第八担当上席 0097
作成日	平成15年 4月10日

<認定情報・付加情報>

【提出日】	平成15年 4月 9日
-------	-------------

次頁無

【書類名】 出願人名義変更届  
【あて先】 特許庁長官殿  
【事件の表示】  
【出願番号】 特願2003-105762  
【承継人】  
【識別番号】 802000031  
【氏名又は名称】 財団法人北九州産業学術推進機構  
【代表者】 有馬 朗人  
【承継人代理人】  
【識別番号】 100121371  
【弁理士】  
【氏名又は名称】 石田 和人  
【電話番号】 093-695-3113  
【手数料の表示】  
【予納台帳番号】 222495  
【納付金額】 4,200円  
【提出物件の目録】  
【物件名】 承継人であることを証する書面 1  
【援用の表示】 平成15年12月12日提出の手續補足書により補足する。  
【物件名】 委任状 1  
【援用の表示】 平成15年12月12日提出のものを援用する。



## 認定・付加情報

特許出願の番号	特願 2003-105762
受付番号	50302053293
書類名	出願人名義変更届
担当官	古田島 千恵子 7288
作成日	平成16年 3月25日

## &lt;認定情報・付加情報&gt;

【提出日】 平成15年12月12日

## 【承継人】

【識別番号】 802000031

【住所又は居所】 福岡県北九州市若松区ひびきの2番1号

【氏名又は名称】 財団法人北九州産業学術推進機構

【承継人代理人】 申請人

【識別番号】 100121371

【住所又は居所】 福岡県北九州市若松区ひびきの2番1号 北九州  
学術研究都市産学連携センターT-302 石田  
特許事務所内

【氏名又は名称】 石田 和人

特願 2003-105762

出願人履歴情報

識別番号 [502428467]

1. 変更年月日 2002年11月26日  
[変更理由] 新規登録  
住所 福岡県福岡市中央区今川1丁目10-43-403  
氏名 笹尾 勤
2. 変更年月日 2003年 5月25日  
[変更理由] 住所変更  
住所 福岡県福岡市中央区鳥飼3丁目15-17  
氏名 笹尾 勤

特願 2003-105762

出願人履歴情報

識別番号

[802000031]

1. 変更年月日

2002年 4月19日

[変更理由]

新規登録

住所

福岡県北九州市若松区ひびきの2番1号

氏名

財団法人北九州産業学術推進機構